The University of Jordan

Authorization Form

I, Nesreen Adnan Hamad, authorize the University of Jordan to supply copies of my Thesis/ Dissertation to libraries or establishments or individuals on request, according to the University of Jordan regulations.

Signature:

Date: 27/4/2011

الجامعة الأردنيـــــة
كلية الدراسات العليا

التاريـــخ: / /

نموذج رقم (١٨)
اقرار والتزام بالمعايير الأخلاقية والأمانة العلمية
وقوانين الجامعة الأردنية وأنظمتها وتعليماتها
لطلبة الماجستير

أنا الطالب: [illegible] الرقم الجامعي: (8020204)

تخصص: علم الحاسوب الكليـــــة: الملك عبدالله الثاني لتكنولوجيا المعلومات

عنوان الرسالة: A clinical decision support system for heart disease diagnosis

اعلن بأنني قد التزمت بقوانين الجامعة الأردنية وأنظمتها وتعليماتها وقراراتها السارية المفعول المتعلقة باعداد رسائل الماجستير عندما قمت شخصيا" باعداد رسالتي وذلك بما ينسجم مع الأمانة العلمية وكافة المعايير الأخلاقية المتعارف عليها في كتابة الرسائل العلمية. كما أنني أعلن بأن رسالتي هذه غير منقولة أو مستلة من رسائل أو كتب أو أبحاث أو أي منشورات علمية تم نشرها أو تخزينها في أي وسيلة اعلامية، وتأسيسا" على ما تقدم فانني أتحمل المسؤولية بأنواعها كافة فيما لو تبين غير ذلك بما فيه حق مجلس العمداء في الجامعة الأردنية بالغاء قرار منحي الدرجة العلمية التي حصلت عليها وسحب شهادة التخرج مني بعد صدورها دون أن يكون لي أي حق في التظلم أو الاعتراض أو الطعن بأي صورة كانت في القرار الصادر عن مجلس العمداء بهذا الصدد.

توقيع الطالب: .... التاريخ: ٢٧ / ٤ / ٢٠١١

A CLINICAL DECISION SUPPORT SYSTEM FOR HEART DISEASE DIAGNOSIS

By
Nesreen Adnan Hamad

Supervisor
Dr. Mohammad Alshraideh

Co-Supervisor
Dr. Mahmoud Abu-Abeeleh

Co-Supervisor
Dr. Moaath Alsmady

This Thesis was submitted in Partial Fulfillment of the Requirements for the Master's Degree of Science in Computer Science

Faculty of Graduate Studies

The University of Jordan

April, 2011

COMMITTEE DECISION

This Thesis /Dissertation (A Clinical Decision Support System For Heart Disease Diagnosis) was successfully Defended and Approved on 17/4/2011.

Examination Committee	Signature
Dr. Mohammad Alshraideh (Supervisor) Assistant Professor of Software Testing Using Evolutionary Algorithms	
Dr. Mahmoud Abu-Abeeleh (Co-Supervisor) Associate Professor of Medicine	
Dr. Moaath Alsmady (Co-Supervisor) Associate Professor of Medicine	
Dr. Saleh Al-sharaeh (Member) Professor of Parallel Processing and High Performance Computing, and Wireless Networks	
Dr. Basel Mahafzah (Member) Associate Professor of Parallel and Distributed Computing, and Interconnection Networks	
Dr. Ibrahim Obeidat (Member) Assistant Professor of Computer Networks (The Hashemite University)	

DIDECATION

To my wonderful parents,

My brothers and sisters,

Hamad's family

And

Abdel’s family

ACKNOWLEDGEMENT

In the name of Allah, the Most Beneficent Most Merciful

All praise is due to **Allah**, the source of all knowledge and strength. I acknowledge His infinite mercy and grace in making this work a success. May His peace and blessings be upon His final messenger **Mohammad**, a guidance and inspiration to our lives.

The successful completion of this work was made possible by a number of major contributions from different persons, to whom I wish to express my gratitude.

I would like to thank:

Dr. Mohammad Alshraideh, thesis supervisor, for the patience guidance, encouragement and advice he has provided throughout my time as his student. I have been extremely lucky to have a supervisor who cared so much about my work, and who responded to my questions and queries so promptly.

Dr. Musa Alakhras for providing fresh ideas and directions in both areas of machine learning and neural networks.

Dr. Basel Mahafzah, for providing helpful advice that improved my scientific research ability during my master's study.

Dr. Mahmoud Abu-Abeeleh, MD, and **Dr. Moaath Alsmady,MD,** thesis Co-Advisors, and the resident doctors who answered my questions about some rudiments medical concepts related to this work.

Nurse Hamdan Alsaaydeh, for his valuable help and suggestions he made in reference to section 3.1 of this work, and for helping me in investigating the ECG findings for all the 162 cases that were used in this thesis.

Ms. Sua'd Rashed, and Mr. Mohammad Albatayneh, for providing the list of patients' names and querying their medical records.

My parents, I would never have been able to pursue this task without their cooperation and understanding. All the knowledge and skills I have today are enhanced by the love they bestowed upon me. I would like to especially thank **my brothers and sisters** for their friendship and support.

Tasneem Abu-Kabeer, **Elham Ahmad**, and **Heba Saadeh**, my dearest colleagues, who encouraged and supported me at the bad and the good moments of this work.

All my friends, who have supported my decisions and have provided vital advice over the years, especially **Rema Maraqa,** my best childhood friend.

Finally, I give many thanks to **those who made it really difficult**; their criticism for choosing this field for my thesis was the greatest motivation to finish this work.

Nesreen A. Hamad

Amman 2011

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Abbreviation	Meaning
AF	Atrial Fibrillation.
ANN	Artificial Neural Network.
ANNs	Artificial Neural Networks.
CAD	Coronary Artery Disease.
CDSS	Clinical Decision Support System.
CDSSs	Clinical Decision Support Systems.
CI	Computational Intelligence.
CVD	Cardiovascular Diseases.
ECG	Electrocardiogram.
GRNN	General Regression Neural Network.
Hardlim	Hard-Limit Transfer Function.
LF	Linear Function.
LM	Levenberg–Marquardt.
MI	Myocardial Infarction.
MLP	Multilayer Perceptron.
NN	Neural Network
PID	Pima Indians Diabetes.
PNN	Probabilistic Neural Network.
RBF	Radial Basis Network.
Rprop	Resilient Back Propagation.

SOM	Self Organizing Maps.
TANH	Hyperbolic Tangent Sigmoid Function.
TMSE	Total Mean Squared Error
UCI	University of California Irvine
USA	Unstable Angina.
WBCD	Wisconsin Breast Cancer Dataset.

A CLINICAL DECISION SUPPORT SYSTEM FOR HEART DISEASE DIAGNOSIS

By
Nesreen Adnan Hamad

Supervisor
Dr. Mohammad Alshraideh

Co-Supervisor
Dr. Mahmoud Abu-Abeeleh

Co-Supervisor
Dr. Moaath Alsmadi

ABSTRACT

Heart disease diagnosis is one of the significant issues all over the world especially in Jordan where mortality records showed that the chief cause of death in Jordan is Cardiovascular Diseases **(CVD)** representing (35.3%) of the total deaths for the year 2007. Significant life saving can be achieved, if an accurate diagnosis is made for patients suffering from various types of heart diseases.

Developing effective Clinical Decision Support Systems **(CDSSs)** involved many researchers in order to improve the ability of the physicians to do their work, where several tools and various methodologies have been proposed to accomplish this goal. One of these methodologies is Artificial Neural Networks **(ANNs),** which are computer paradigms that belong to the Computational Intelligence **(CI)** family and are inspired by the biological neural system.

In this thesis, a Multilayer Perceptron **(MLP)** feed forward neural network is used to develop a Clinical Decision Support System **(CDSS)** for the diagnosis of three heart diseases: Atrial Fibrillation **(AF)**, Myocardial Infarction **(AF)** and Unstable Angina **(USA)**. The input layer of the system includes 58 variables for each patient such as age, gender, pulse, etc. The output layer contains one neuron that will produce a number which represents the heart disease type a patient suffers from. The number of hidden layers and nodes in the hidden layers are determined through an iterative process. Furthermore, a Resilient Back Propagation **(Rprop)** algorithm is used to train the system. In particular, a ten-fold cross validation scheme was used to access the generalization of the proposed system. The experiments were performed with the proposed system. We obtained 89.3% classification accuracy from the experiments made on data taken from 162 patients' medical records suffering from three heart diseases managed at the cardiology clinics at Jordan University Hospital **(JUH)**.

Chapter One

Introduction

1. Introduction

A major class of problems in medical science involves the diagnosis of a disease based upon various symptoms and tests performed upon the patient. When several symptoms and tests are involved, the ultimate diagnoses may be difficult to obtain, even for a medical expert. Furthermore, similar symptoms for multiple diseases may boost the chance for a medical misdiagnosis. Consequently, over the past few decades this has given rise to computerized diagnostic tools proposed to assist the physician to clarify the confusing data in case of imprecision and uncertainty (Kiyan and Yildirim, 2003), these computerized tools are called Clinical Decision Support Systems **(CDSSs)** which are types of computer programs that can be developed through several kinds of machine learning techniques, and are used to assist clinicians at the point of care (Yan et al., 2006).

In this chapter, section 1.1 illustrates a general overview of machine learning, section 1.2 concerns about CDSSs, section 1.3 gives a general overview of heart diseases, section 1.4 introduces the problem statement and thesis motivation, and thesis organization is illustrated in section 1.5.

1.1 Machine Learning Overview

Machine learning studies computer algorithms for learning to do stuff, learning to complete a task, or to make accurate predictions, or to behave intelligently. The learning that is being done is always based on some sort of observations or data such as examples (the case in this thesis) and direct experiences or instructions. So in general, machine

learning is about learning to do better in the future based on what was experienced in the past (Mitchell, 1997).

Learning, like intelligence, covers such a broad range of processes that it is difficult to define precisely. A dictionary definition includes phrases such as "to gain knowledge, or understanding of, or skill in, by study, instruction, or experience," and "modification of a behavioral tendency by experience." Zoologists and psychologists study learning in animals and humans (Mitchell, 1997). There are several parallels between animals and machine learning. Certainly, many techniques in machine learning were derived from the efforts of psychologists to make more precise their theories of animal and human learning through computational models. It seems likely also that the concepts and techniques being explored by researchers in machine learning may illuminate certain aspects of biological learning (Fausett, 2001).

As regards machines, we might say, very broadly, that a machine learns whenever it changes its structure, program, or data (based on its inputs or in response to external information) in such a manner that it is expected future performance improves. Some of these changes, such as the addition of a record to a data base, fall comfortably within the area of other disciplines and are not necessarily better understood for being called learning. But, for example, when the performance of a speech-recognition machine improves after hearing several samples of a person's speech, we feel quite justified in that case saying that the machine has learned(Veropoulos, 2001).

One might ask "Why should machines have to learn?" There are several reasons why machine learning is important. Of course, we have already mentioned that the achievement of learning in machines might help us understand how animals and humans learn. But there are important engineering reasons as well (Mitchell, 1997). Some of these are:

- Some tasks cannot be defined well except by example; that is, we might be able to specify input/output pairs but not a concise relationship between inputs and desired outputs. We would like machines to be able to adjust their internal structure to produce correct outputs for a large number of sample inputs and thus suitably constrain their input/output function to approximate the relationship implicit in the examples(Fausett, 2001).
- The amount of knowledge available about certain tasks might be too large for explicit encoding by humans. Machines that learn this knowledge gradually might be able to capture more of it than humans would want to write down.
- Environments change over time. Machines that can adapt to a changing environment would reduce the need for constant re-design.
- New knowledge about tasks is constantly being discovered by humans. Continuing re-design of AI systems to conform to new knowledge is impractical, but machine learning methods might be able to track much of it.

Machine learning techniques have been employed by many disciplines to automate complex decision making and problem solving tasks. This field of research is immensely diverse, with applications in medicine, mathematics, computer science, chemistry, economics, business management and many other fields (Ramalingam et al., 2006).The majority of these techniques are concerned in pattern recognition problems. Pattern recognition is the act of taking in raw data and taking an action based on the "category" or the pattern (Duda et al., 2001). An example of pattern recognition problems is classification (Mitchell, 1997). In machine learning, the solution of pattern recognition problems lies within the field of supervised learning (the base for this thesis). The task is to learn (induce)

the relationship between the dependent attributes (input) and the designated attribute (output) from a set of examples, i.e. generalize information collected from given data to unseen data. A common supervised machine learning tools are: Multilayer Perceptron (MLP) neural network (Yan et al., 2006) and decision trees (Mitchell, 1997). Another category of problems in machine learning is unsupervised learning. Unsupervised learning is a class of problems in which one seeks to determine how the data are organized. It is distinguished from supervised learning in that the learner is given only unlabeled examples (Geoffrey and Terrence,1999). One form of unsupervised learning is clustering which is the assignment of a set of observations into subsets (called clusters) so that observations in the same cluster are similar in some sense among neural network models. The Self-organizing map **(SOM)** is an example of unsupervised machine learning tool (Geoffrey and Terrence, 1999).

1.1.1 Applications of Machine Learning

Because of their ability to identify patterns or trends in data, machine learning methods have been successfully applied in many industries review of (Fausett, 2001):

- **Spoken language understanding:** within the context of a limited domain, by determining the meaning of something uttered by a speaker to the extent that it can be classified into one of a fixed set of categories.
- **Spam filtering:** identify email messages as spam or non-spam.
- **Credit card fraud detection:** identify credit card transactions (for instance) which may be fraudulent in nature.
- **Weather prediction:** predict, for instance, whether or not it will rain tomorrow.

- **Medical diagnosis:** diagnose a patient as a sufferer or non-sufferer of some diseases.

1.1.2 Machine Learning in the Medical Domain

Machine learning algorithms were from the very beginning designed and used to analyze medical data sets. Today machine learning provides several indispensable tools for intelligent data analysis. Especially in the last few years, the digital revolution provided relatively inexpensive and available means to collect and store the data. Modern hospitals are well equipped with monitoring and other data collection devices, and data is gathered and shared in large information systems (Veropoulos, 2001). Machine learning technology is currently well suited for analyzing medical data, and in particular there is a lot of work done in medical diagnosis in small specialized diagnostic problems. Data about correct diagnoses are often available in the form of medical records in specialized hospitals or their departments. All that has to be done is to input the patient records with known correct diagnosis into a computer program to run a learning algorithm (Fausett, 2001). This is of course an oversimplification, but in principle, the medical diagnostic knowledge can be automatically derived from the description of cases solved in the past. The derived classifier (usually called CDSS) can then be used either to assist the physician when diagnosing new patients in order to improve the diagnostic speed, accuracy and/or reliability, or to train students or physicians non-specialists to diagnose patients in a special diagnostic problem (Mitchell, 1997).

1.2 Clinical Decision Support Systems Overview

In medical diagnosis, some of the scopes for ambiguity in the inputs are: the history (patient's description of the disease condition), physical examinations (especially in cases of uncooperative patients) and laboratory tests (faulty methods or equipment). Moreover, in the treatment, there are chances of drug reactions and specific allergies (Brause, 2001). All these factors may not be taken into account by a busy clinician attending hundreds of patients daily. Therefore, computers can help the clinician to reach an accurate diagnosis faster because they make fewer errors than humans in making analyses of complex data (Rasheed, 2009). As a result, CDSSs were devised.

In the literature many researchers have given their definitions of CDSSs; Musen (1997) defined a CDSS as ‘’ any piece of software that takes information about a clinical situation as inputs and that produces inferences as outputs that can assist practitioners in their decision making and that would be judged as “intelligent” by the program users’’ . Miller and Geissbuhler (2007) defined a CDSS providing diagnostic decision support as a ‘‘computer-based algorithm that assists a clinician with one or more component steps of the diagnostic process’’. Sim as cited in (kong et al., 2008) defined CDSS as ‘’ software that was designed to be a direct aid to clinical decision-making, in which the characteristics of an individual patient are matched to a computerized clinical knowledge base and patient specific assessments or recommendations are then presented to the clinician or the patient for a decision’’. So in general CDSSs are computer programs designed to help healthcare professionals to make clinical decisions. In a sense, any computer system that deals with clinical data or knowledge is intended to provide decision support. These systems can be

used for diagnosis, treatment of health diseases, and future evaluation of the patient (Kong et al., 2008).

The implementation of these systems could have several beneficial effects; they improved patient safety by for example through reducing medication errors and adverse events; medical errors imply actions such as the conclusion of wrong diagnosis, the prescription of an incorrect treatment or the inefficient execution of correct treatment. Additionally, CDSSs improved clinical documentation and patient satisfaction. Moreover, they supported medical education and training and provided immediate feedback to patients (Thornett, 2001). Because of these benefits, CDSSs were applied in many medical domains, such as cardiology, gynecology, diabetes, etc (Veropoulos, 2001). In this thesis we will focus on the cardiology domain.

1.3 Heart Diseases Overview

Heart disease is a general term that refers to a variety of acute and chronic medical conditions that affect one or more of the components of the heart. The heart is a muscular, fist-sized organ that is located in the left side of the chest cavity. It continuously pumps blood, beating as many as 100,000 times a day. The blood that the heart moves carries oxygen and nutrients throughout the body and transports carbon dioxide and other wastes to the lungs, kidneys, and liver for removal. The heart ensures its own oxygen supply through a set of coronary arteries. Therefore, anything that damages the heart or decreases the heart's supply of oxygen, makes it less efficient, reduces its ability to fill and pump, will disrupt the coordinated relationship between the heart, kidneys, and blood vessels and will harm not only the heart but the rest of the body as well (Tanser, 2006).

Coronary Artery Disease **(CAD)** is the most common form of heart disease. It is usually part of a systemic Cardiovascular Diseases **(CVD)**, and it is caused by a narrowing of arteries in the heart and throughout the body over time due to a build-up of fatty deposits that form plaques. This narrowing can significantly limit the amount of blood carried by the arteries and decrease the amount of oxygen supplied to the tissues. As the coronary blood vessels gradually narrow, symptoms do not appear until critical decrease in blood flow to an area of the heart, when it can cause intermittent chest pain (angina) upon exercise that worsens in frequency and severity over time (Toscano, 2001). When plaques suddenly ruptures (termed unstable plaques), an acute narrowing of the coronary artery can cause chest pain to develop at rest or with minimum exertion (termed Unstable Angina **(USA)**) or can even cause death of an area of the myocardium or heart attack (termed Myocardial Infarction **(MI)**) (Williams et al., 1999).

Other signs and complications that may arise from heart diseases includes; Arrhythmias which are changes in the rhythm of heartbeats or in the strength of heart contractions, the most common type of Arrhythmias is Atrial Fibrillation **(AF)**, where the result from an AF is an irregular heartbeat, which may occur in episodes lasting from minutes to weeks, or it could occur all the time for years. It may result in palpitations, fainting, or chest pain. The natural tendency of AF is to become a chronic condition. Chronic AF leads to a small increase in the risk of death (Benjamin et al., 1998).

Heart diseases have a variety of signs and symptoms that frequently change and/or worsen over time. Chronic heart diseases can have episodes with acutely worsened symptoms; these may resolve (either on their own or with treatment), persist, or even become life-threatening. Patients with early heart disease may experience few or vague symptoms, such as fatigue, shortness of breath with or without effort, dizziness, and/or

nausea; however, these symptoms do not indicate the particular type of heart disease present. These symptoms may also be seen with a variety of other conditions. In order to distinguish between symptoms that are heart-related and those that are due to another condition; testing is ordered to help in determining which heart disease is present, to determine whether the disorder is acute or chronic, to monitor a cardiac event that is in progress such as a heart attack, and to determine the severity and extent of the disease (Desvigne-Nickens, 2007).

A test may include:

- A medical history, including an evaluation of risk factors such as age, CAD, diabetes, and smoking,
- A physical examination.
- Laboratory blood tests such as cardiac biomarkers proteins (i.e. Troponin and CK-MB) that are released when muscle cells are damaged, are frequently ordered when patients have symptoms of acute coronary syndrome, such as chest pain, pain in the jaw, neck, abdomen, back, or that radiates to the shoulder or arms, nausea, and dyspnea.
- An Electrocardiogram **(ECG)** which is a test that looks at the heart's electrical activity and rhythm.

1.4 Problem Statement and Thesis Motivation

Statistics have consistently shown that heart diseases are one of the leading causes of deaths in the US and all over the world where heart diseases in the US represent 24.9% of the total deaths (Miniño et al., 2010). The case here in Jordan is the same, where

mortality records showed that the primary cause of death is CVD with a percentage of 35.2% (Dababneh et al., 2010).

Moreover, Brause (2001) showed in his study that machine learning can help in making correct diagnoses by reducing medical errors where the number of deaths caused by medical error has been estimated to be over 98.000 per year. Having an effective CDSS will significantly reduce this number.

From these results this thesis aims to develop a CDSS based on the patients' clinical records that were collected from the cardiology clinics at Jordan University Hospital **(JUH)** for three types of heart diseases; USA, MI and AF, in order to accomplish this, a neural network with a resilient back propagation will be used. This thesis will also investigate the number of hidden neurons and hidden layers to observe its impact on the classification accuracy.

1.5 Thesis Organization

This thesis is organized as follows: preliminaries concepts and background information related to ANN and some literature review are presented in Chapter 2. Data preparation and system architecture are illustrated in Chapter 3. The experimental results of different architectures that were applied on the collected data sets are presented in Chapter 4. Chapter 5 concludes this thesis and proposes some future works.

Chapter Two

Literature Review

2. Literature Review

2.1 Artificial Neural Networks Background

Computational Intelligence **(CI)** is a set of nature-inspired computational methodologies and approaches to address complex problems of the real world applications to which traditional methodologies and approaches are ineffective or infeasible (Thammano and Moolwong, 2009). One of the techniques that belong to the CI family is Artificial Neural Networks **(ANNs)** (usually called Neural Network **(NN)**) that were introduced in 1940s (further developed in 1980s) as an attempt to emulate the features of biological neural networks in order to address a range of difficult information processing, analysis and modeling problems (Fausett, 2001). They are a powerful data modeling tool that is able to capture and represent complex input/output relationships. Since the motivation for the development of NN technology stemmed from the desire to develop an artificial system that could perform "intelligent" tasks similar to those performed by the human brain; as a result neural networks resemble the human brain in the following two ways:

1. A neural network acquires knowledge through learning (experience).

2. A neural network's knowledge is stored within inter-neuron connection strengths known as weights.

ANNs have seen an explosion of interest over the last few years, and are being successfully applied across an extraordinary range of problem domains, in areas as diverse as finance, medicine, engineering, geology and physics (Mehrotra et al., 1996). Indeed, anywhere that there are problems of prediction or classification, neural networks are being introduced (Haykin, 2001). This success can be attributed to a few key factors:

- **Power:** Neural network models can implicitly detect complex nonlinear (a term which is discussed in more details later in subsection 2.1.4.3) relationships between independent and dependent variables, also they have the ability to detect all possible interactions between predictor variables and can be developed using multiple different training algorithms (Fausett, 2001).
- **Ease of use:** Neural networks learn by example. The neural network user gathers representative data, and then invokes training algorithms to automatically learn the structure of the data (Haykin, 2001). Although the user does need to have some heuristic knowledge of how to select and prepare data, how to select an appropriate neural network and how to interpret the results.

Neural networks are applicable in virtually every situation in which a relationship between the predictor variables (independents, inputs) and predicted variables (dependents, outputs) exists, even when that relationship is very complex and not easy to articulate in the usual terms of "correlations" or "differences between groups" (Fausett, 2001). A few representative examples of problems to which neural network analysis has been applied successfully are:

- **Credit assignment:** A variety of pieces of information are usually known about an applicant for a loan. For instance, the applicant's age, education, occupation, and many other facts may be available. After training a neural network on historical data, neural network analysis can identify the most relevant characteristics and use those to classify applicants as good or bad credit risks (Haykin, 2001).
- **Monitoring the condition of machinery:** Neural networks can be instrumental in cutting costs by bringing additional expertise to scheduling the preventive

maintenance of machines (Mehrotra et al., 1996). A neural network can be trained to distinguish between the sounds a machine makes when it is running normally ("false alarms") versus when it is on the verge of a problem. After this training period, the expertise of the network can be used to warn a technician of an upcoming breakdown, before it occurs and causes costly unforeseen "downtime".

Due to the importance of ANNs as discussed above, the next subsections will concern about giving a general background of the biological model that inspired the development of ANNs and comparing it with the basic model of an Artificial Neural Network **(ANN)**. The activation functions that are commonly used when building an ANN model are also discussed. Beside the common two types of ANNs and the training algorithms, the designing issues that are related to ANNs are also presented.

2.1.1. The Biological Model

The basic model of the artificial neuron is founded upon the functionality of the biological neuron. By definition, "Neurons are basic signaling units of the nervous system of a living being in which each neuron is a discrete cell whose several processes are from its cell body" (Haykin, 2001). Each neuron is a specialized cell which can propagate an electrochemical signal. The neuron has a branching input structure (the dendrites), a cell body, and a branching output structure (the axon). The axons of one cell connect to the dendrites of another via a synapse as shown in Figure 2.1. When a neuron is activated, it fires an electrochemical signal along the axon. This signal crosses the synapses to other neurons, which may in turn fire. A neuron fires only if the total signal received at the cell body from the dendrites exceeds a certain level (the firing threshold). The strength of the

signal received by a neuron (and therefore its chances of firing) critically depends on the efficacy of the synapses. Each synapse actually contains a gap, with neurotransmitter chemicals poised to transmit a signal across the gap junction (Haykin, 2001).

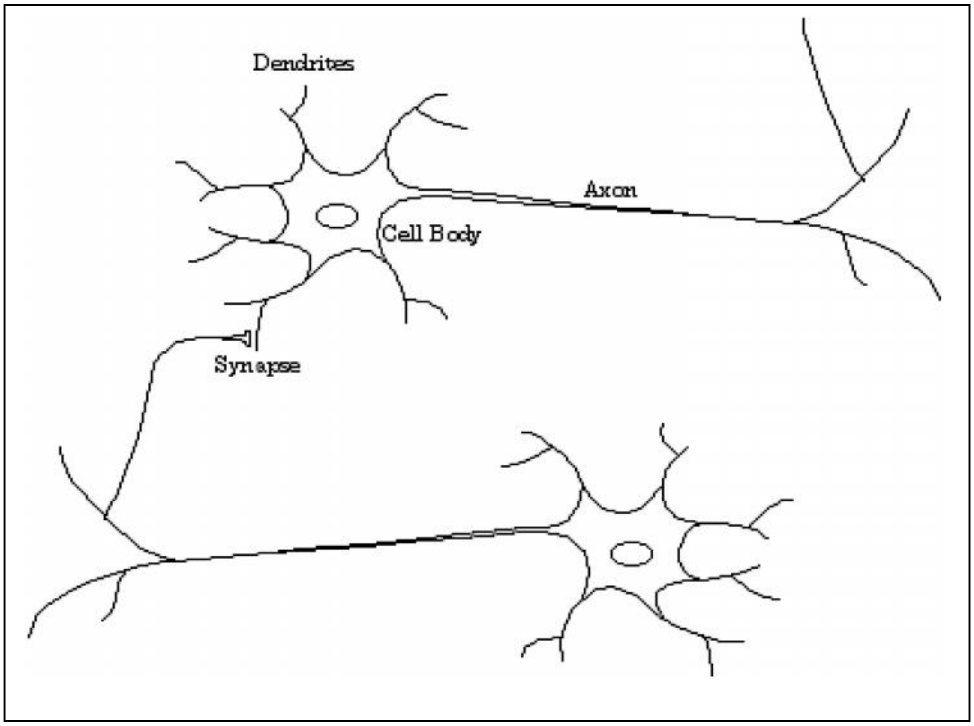


Figure 2.1: Biological Neuron; (Haykin, 2001)

The brain which is principally composed of a very large number (10,000,000,000) of neurons, massively interconnected (with an average of several thousand interconnects per neuron, although this varies enormously) will analyze all patterns of signals sent, and from that information it interprets the type of information received. Thus, from a very large number of neurons (each performing a weighted sum of its inputs, and then firing a binary signal if the total input exceeds a certain level) the brain manages to perform extremely complex tasks (Haykin, 2001). Of course, there is a great deal of complexity in the brain

which has not been discussed here, but it is interesting that ANNs can achieve some remarkable results using a model not much more complex than this.

2.1.2. The Basic Model

The basic model of ANNs is an input layer consisting of nodes that simply accept the input values. The outputs of neurons in a layer are inputs to neurons in the next layer. The last layer is called the output layer. Layers between the input and output layers are known as hidden layers. Figure 2.2 illustrates a diagram for this architecture.

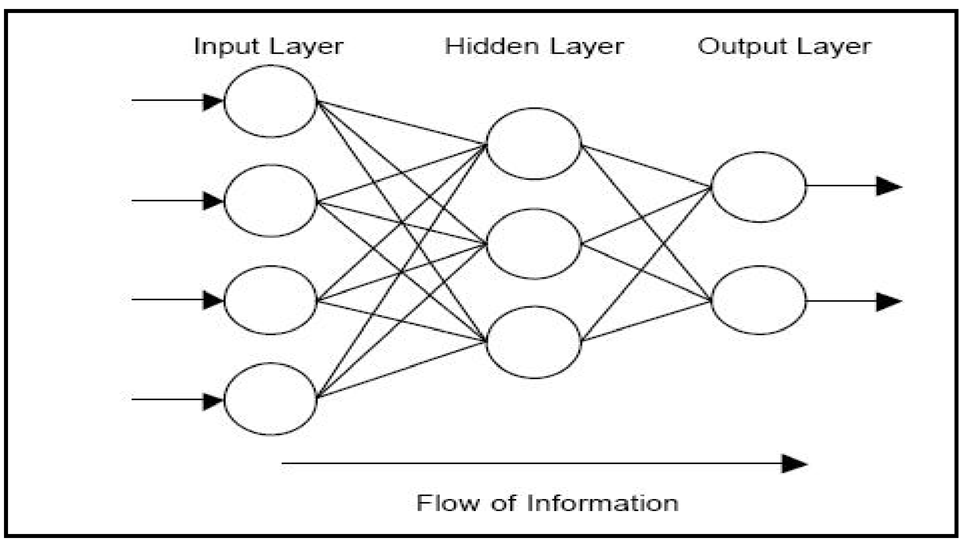


Figure 2.2: The basic model of an ANN

To capture the essence of biological neural systems, an artificial neuron is defined (Fausett, 2001) as follows:

- It receives a number of inputs (either from original data, or from the output of other neurons in the neural network). Each input comes via a connection that has a

strength (or weight); these weights correspond to synaptic efficacy in a biological neuron.

- The weighted w_i sum of the inputs u_i is formed to compose the activation of the neuron. Gathering the weighs of the inputs according to:

$$Sum = \sum_{i=1}^{n} u_i w_i$$

- The activation signal is passed through an activation function *(y)* (also known as a transfer function) to produce the output of the neuron as follows:

$$y = f\left(\sum_{i=1}^{n} u_i w_i\right)$$

Figures 2.3 illustrates a summary of the principle work of an artificial neuron model, and a schematic illustrations of the basic features of a biological neuron and its artificial basic model is illustrated in Figure 3.4

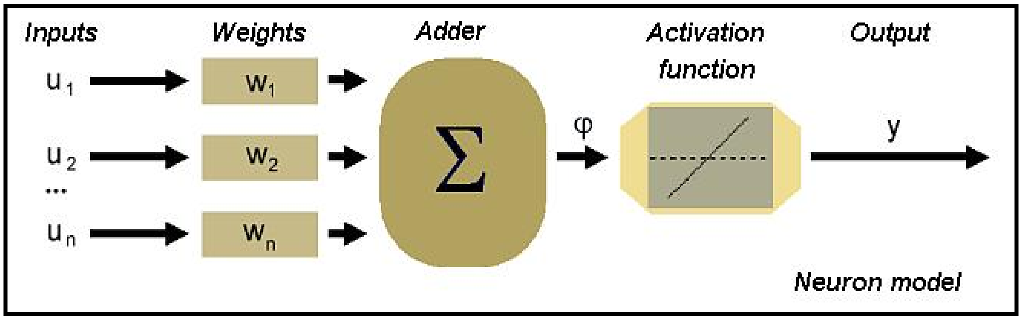


Figure 2.3: Summary of the principle work of Artificial Neuron Model; (Fausett, 2001)

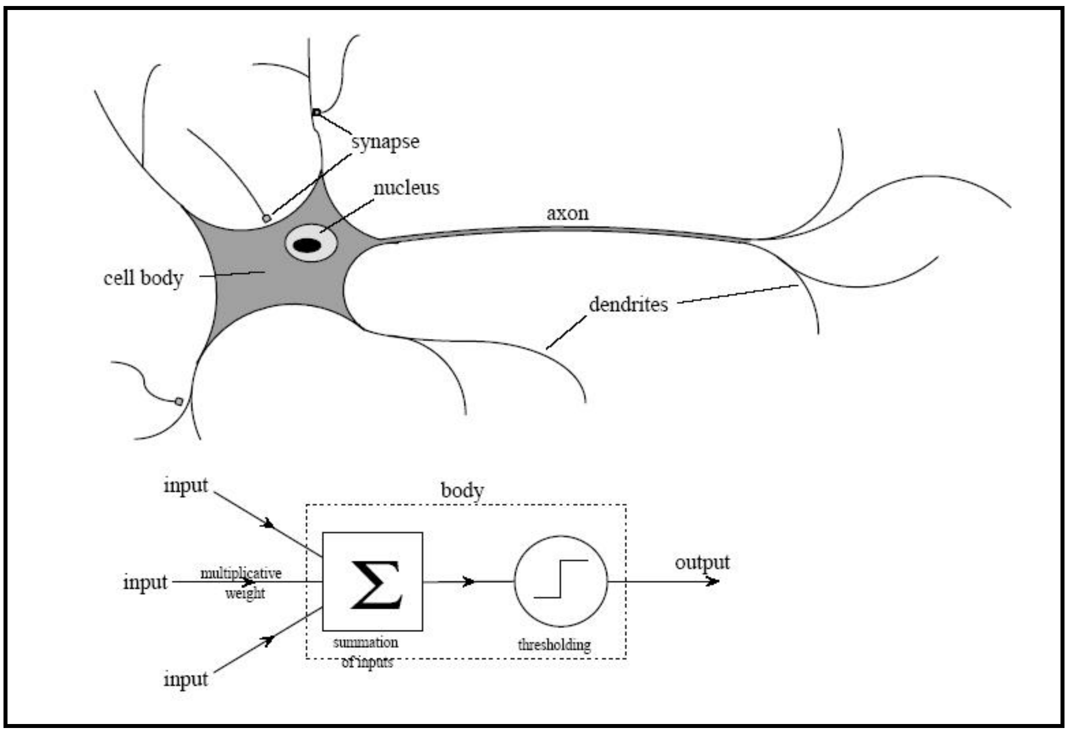


Figure 2.4 : Schematic Illustrations of the Basic Features of a Biological Neuron and its Artificial Basic Model; (Fausett, 2001)

2.1.3. Types of Activation Function

The output of a particular artificial neuron is determined by the type of activation function, as known sometimes transfer function (Freeman and Skapura, 1991).An activation function is used to limit the amplitude of the output of a neuron and is commonly known as a squashing function (Haykin, 2001). Neural Networks supports a wide range of activation functions. The most common types of activation functions are:

1. **Linear activation function (LF)**: The outputs of a linear activation function, also known as a linear combiner, are determined as follows: ***a*= *n*** where ***n*** is the sum of the weighted inputs. The graph of a LF function is represented as in Figure 2.5

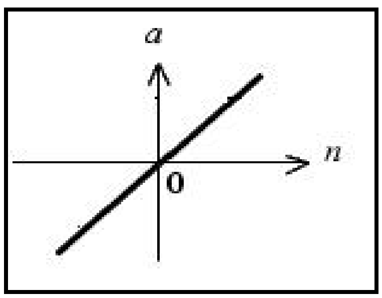


Figure 2.5: Graph of a Linear Activation Function

2. **Hyperbolic tangent sigmoid activation function (TANH)**: The hyperbolic tangent sigmoid activation function has the following formula: $a = \frac{2}{1 + e^{-2n}} - 1$

where *n* is the sum of the weighted inputs. The resulted output from this function will be in a range between [-1,1] (Hagan et al., 1996). The graph of a TANH function is represented as in Figure 2.6

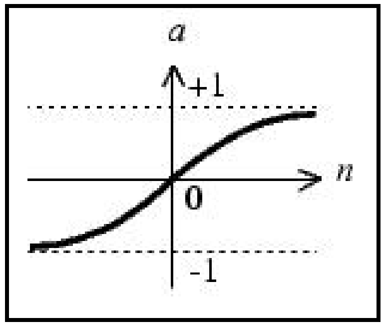


Figure 2.6: Graph of a Hyperbolic Tangent Sigmoid Activation Function

2.1.4. ANNs Architectures

ANNs architecture can be divided into two main categories based on their processing structure (known as topology): *feedback and feed-forward* neural networks.

In feedback neural networks, signals can travel in both directions. Thus the network has loops in its topology as shown in Figure 2.7(a). Networks with this structure can be very complicated and powerful, as they can continuously change state until an equilibrium is reached – this balanced state is then kept up until changes in the input data occur (Mehrotra et al., 1996).

In feed-forward neural networks signals are allowed to travel one way only; from input to output as shown in Figure 2.7 (b). There are no feedback (loop) connections. Feed-forward networks tend to be straight forward models that associate inputs with outputs (Mehrotra et al., 1996).

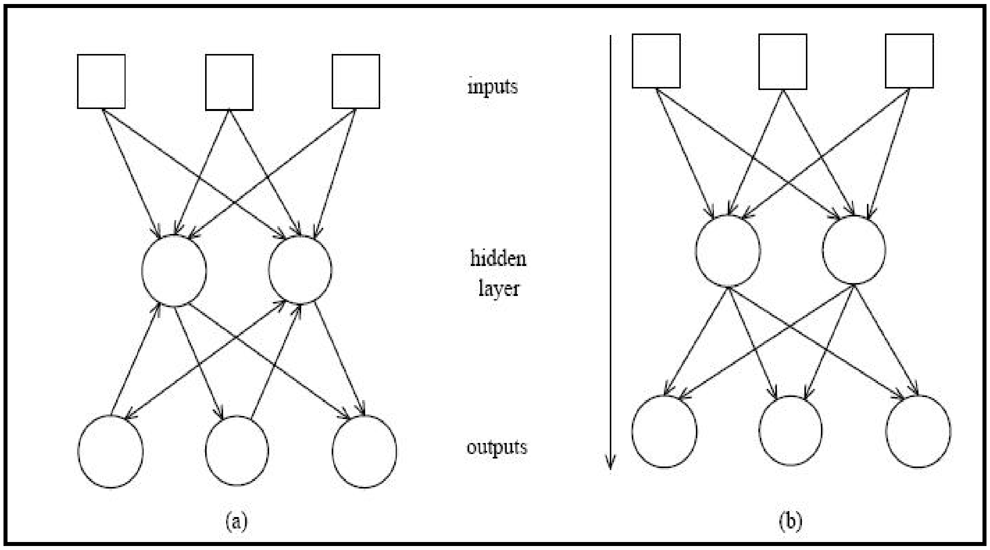


Figure 2.7: A (a) Feedback and a (b) Feed-forward Neural Network.

These architectures can be further subdivided into two main classes based on the number of layers of processing nodes used in the model; single-layer perceptron and multilayer perceptron (Fausett, 2001). In a single-layer perceptron there is a layer of input nodes and a layer of output nodes, which are the only processing nodes in the model. In a multilayer perceptron (MLP) there are one or more hidden layers between the input and the output nodes.

2.1.4.1 The Single –Layer Perceptrons

In Single-Layer Perceptrons, or Perceptron, there is a layer of input nodes and a layer of output nodes, which are the only processing nodes in the model (Krose and Smagt, 1996). The general perceptron network architecture is shown in Figure 2.8.

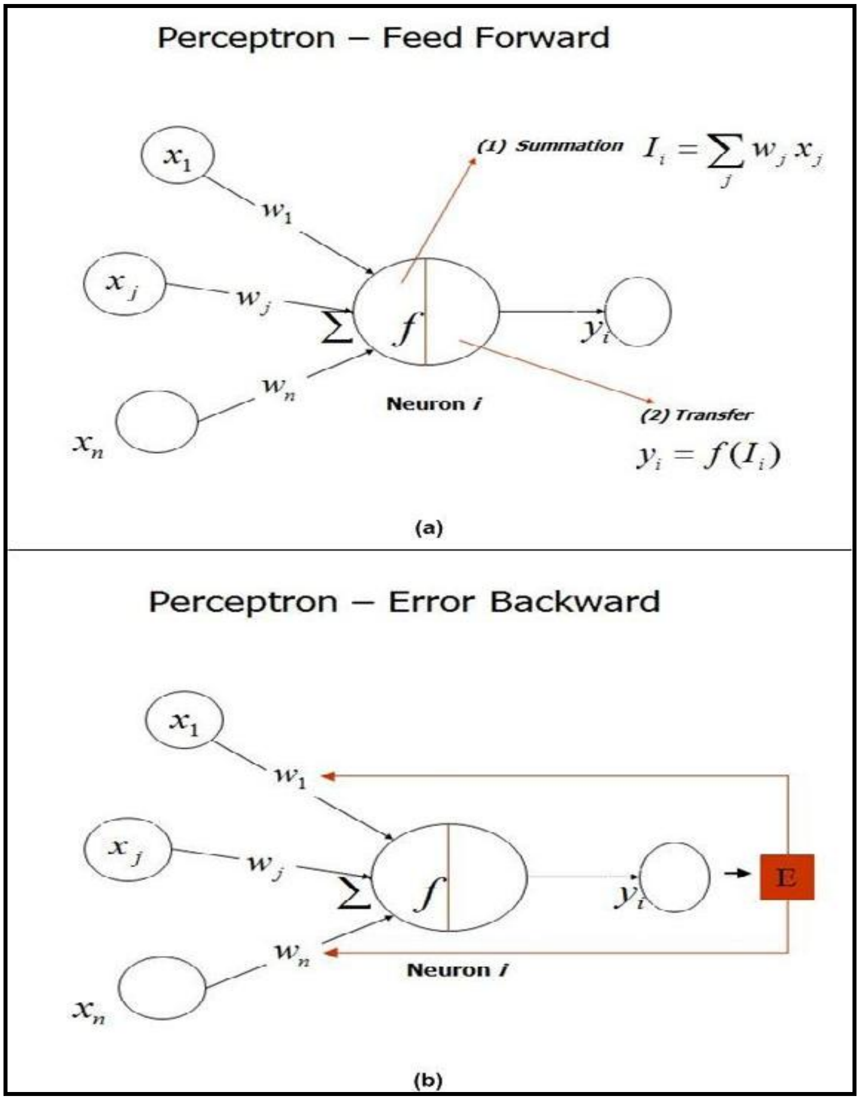


Figure 2.8: Feed-Forward Network with Single-Layer Perceptron (a) Feed Forward Stage, (b) Error Backward Stage

This network is one of the simplest ANNs whose weights and biases could be trained to produce a correct target vector presented with the corresponding input vector (Veropoulos, 2001). Perceptrons are especially suited for simple problems in pattern classification (linearly separable problems).This network is often trained with the perceptron learning rule; the perceptron learning rule is a learning rule for training single–layer hard–limit networks.

A perceptron neuron uses the hard-limit (threshold) transfer function, the hard-limit transfer function is one of the transfer functions that maps inputs greater than or equal to 0 to 1, and all other values to 0 (Freeman and Skapura ,1991). In a perceptron neuron (which uses the hard-limit transfer function (hardlim)) each external input is weighted with an appropriate weight w_j, and the sum of the weighted inputs is sent to the hard-limit transfer function, which also has an input of 1 transmitted to it through the bias (Krose and Smagt, 1996).

The perceptron rule has two main stages; Feed-forward stage as in Figure 2.8(a), and error backward stage as in Figure 2.8(b), the details for each stage is as follow:

1. **Feed-forward stage**. Perceptron generates initially randomly weights, and then the weights (w_j) and input values x_j are summed to compose activation function (v) as follows (Mehrotra et al., 1996):

$$I_i = w_0 + \sum_{j=1}^{m} w_j x_j$$

Where w_0 is called the bias which is a numerical value associated with the neuron which makes the neuron more powerful. It is convenient to think of the bias as the weight for an input x_0 whose value is always equal to one, so that:

$$I_i = \sum_{j=0}^{m} w_j x_j$$

The perceptron neuron produces a 1 if the net input into the transfer function is equal to or greater than 0; otherwise it produces a 0.

2. **Error backward stage.** In this stage the output error is calculated as the following:

$\varepsilon_i = t_i - y_i$, where t_i is the target and y_i is the output, if the error is equal to zero then no weight update is performed, else the weight must be updated by applying the following:

$$w_j^{(new)} = w_j^{(old)} + \Delta w_j$$

$$\text{where } \Delta w_j = \varepsilon_i \eta x_j$$

The term η is called the learning rate and is used to determine by how many the weights are changed at each step.

2.1.4.2 Multilayer Perceptron (MLP)

In MLP, or multilayer feed-forward networks, there are one or more hidden layers between the input and the output nodes (Machado1996) as shown in Figure 2.2. The nodes belonging to these layers are usually referred to as hidden nodes. A simple network has a feed-forward structure: signals flow from inputs, forwarded through any hidden units, eventually reaching the output units. In other words, the feed-forward network is a layered network in which each layer only receives inputs from previous layers (Mehrotra et al., 1996). The feed-forward structures have proved most useful in solving non-linearly separable problems (Haykin, 2001).

The input layer in an MLP contains units that simply serve to introduce the values of the input variables. The hidden and output layer neurons are each connected to all of the units in the preceding layer (Freeman and Skapura ,1991). When the network is executed (used), the input variable values are placed in the input units, and then the hidden and output layer units are progressively executed. Each of them calculates its activation value

by taking the weighted sum of the outputs of the units in the preceding layer (Fausett, 2001). The activation value is passed through the activation function to produce the output of the neuron. When the entire network has been executed, the outputs of the output layer act as the output of the entire network.

2.1.4.3 Linearly and Non-Linearly Separable Problems

A problem is said to be linearly separable; if a straight line can be drawn to separate the input vectors into two categories, if more than one line is needed to separate the input vectors then we say that this problem is non-linearly separable (Freeman and Skapura ,1991). As was said before, the single layer perceptrons are powerful to be used with linearly separable problems, while MLP are used with non-linearly separable problems.

A common example of a linearly separable problem is the AND Boolean function. Table 2.1 is the truth table of the AND Boolean function, represented as in Figure 2.9, we can notice that the AND operation is a linearly separable problem because the separable line classified three points (0,0), (0,1) and (1,0) as one group, while the point (1,1) was classified as another group.

Table 2.1: Truth Table of the AND Boolean Function

1	**X2**	**X1** *AND* **X2**
0	0	0
0	1	0
1	0	0
1	1	1

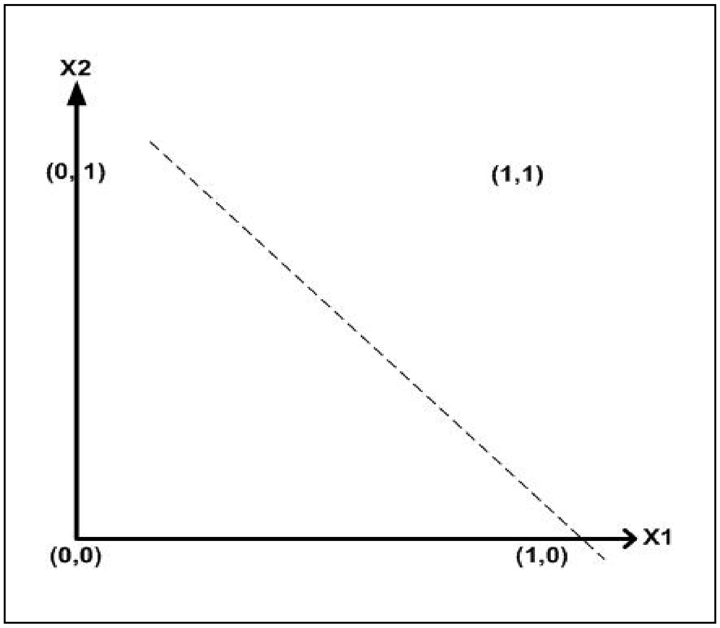


Figure 2.9: The AND Function is Linearly Separable

A common example of a non-linearly separable problem is the Boolean exclusive or (XOR) function, with the truth table shown in Table 2.2 , the desired computation is ‘’yes (1)’’ when one or the other of input is on, and ‘’no (0)’’ when they are both off or both on. So as illustrated in Figure 2.10, there is no clear line that can separate the two types of points.

Table 2.2: Truth Table of the XOR Boolean Function

X1	**X2**	**X1** *XOR* **X2**
0	0	0
0	1	1
1	0	1
1	1	0

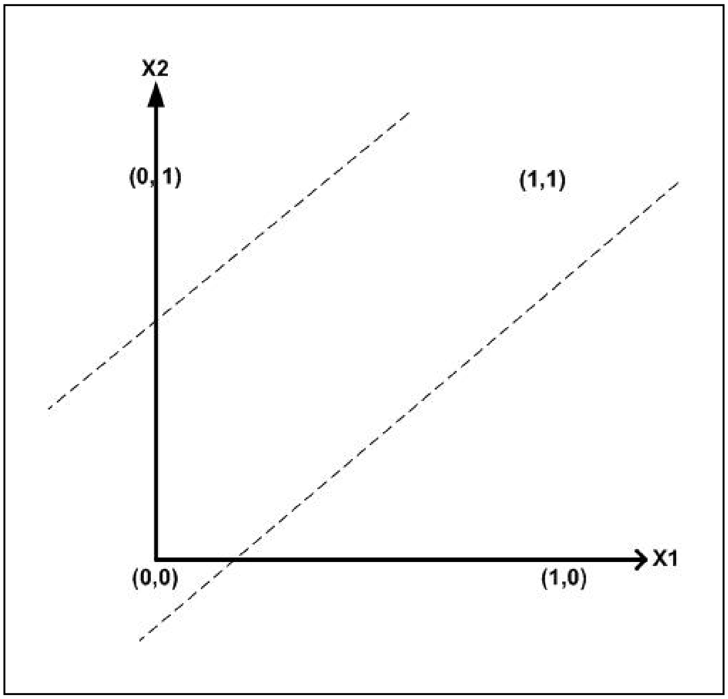


Figure 2.10: The XOR Function is Non-Linearly Separable

2.1.5. The Back Propagation Algorithm

MLP neural networks have been applied successfully to solve difficult and diverse problems by training them in a supervised manner with highly popular algorithm known as Back Propagation which uses the data to adjust the network's weights and biases in manner that minimizes the error in its predictions on the training set (Krose and Smagt, 1996). If the network is properly trained, it has then learned to model the (unknown) function that relates the input variables to the output variables, and can subsequently be used to make predictions where the output is not known.

A Back Propagation network learns by example. You give the algorithm examples of what you want the network to do and it changes the network's weights so that, when training is finished, it will give you the required output for a particular input. In order to

train the network you need to give it examples of what you want (the output you want (called the *Target*)) for a particular input as shown in Figure 2.11.

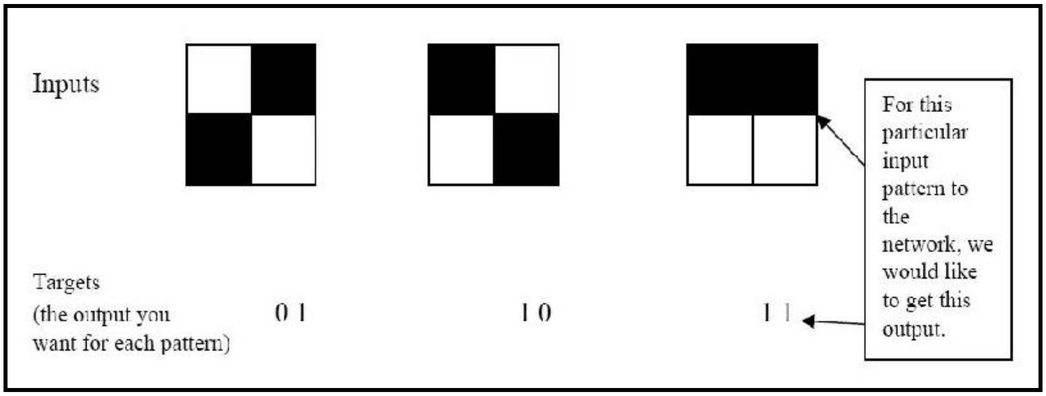


Figure 2.11: A Back Propagation Training Set; (Krose and Smagt, 1996)

So, if we put in the first pattern to the network, we would like the output to be 0 1 as shown in Figure 2.12 (a black pixel is represented by 1 and a white by 0). The input and its corresponding target are called a *Training Pair*. Once the network is trained, it will provide the desired output for any of the input patterns.

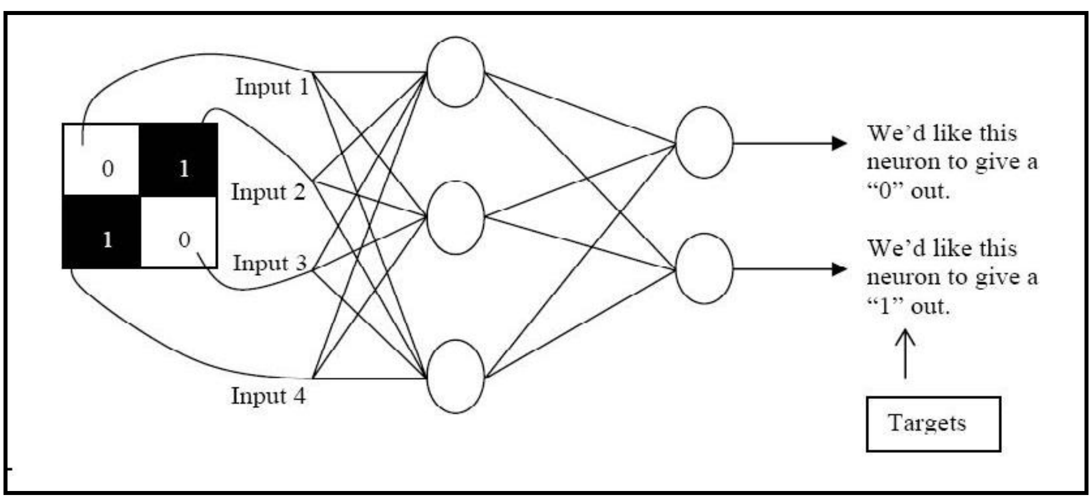


Figure 2.12: Applying a Training Pair to a Network; (Krose and Smagt, 1996)

The training works as follows (Krose and Smagt, 1996):

- The network is first initialised by setting up all its weights to be small random numbers.
- Next, the input pattern is applied and the output calculated (this is called the *forward pass*). The calculation gives an output which is completely different to what you want (the Target), since all the weights are random.
- We then calculate the *Error* of each neuron, which is essentially:
  Target –Output (i.e What you want – What you actually get).
- This error is then used mathematically to change the weights in such a way that the error will get smaller. In other words, the Output of each neuron will get closer to its Target (this part is called the *reverse pass*).
- The process is repeated again and again until the error is minimal.

2.1.5.1 The Error Surface

Since Back Propagation iteratively searches for a set of weights that minimizes the error function **(E)** over all training pairs. Therefore, MLP networks have an error surface, which is in general complex and believed to have many local and global minima. This occurs because the algorithm always changes the weights in such a way as to cause the error to fall. But the error might briefly have to rise, as shown in Figure 2.13. If this is the case, the algorithm will get stuck (because it can't go uphill) and the error will not decrease further (Fausett, 2001). This requires rerunning the algorithm with new random initial weights and other network parameter hoping to reach the global minima in the new run. As a consequence, Back Propagation is never assured to find the global minima.

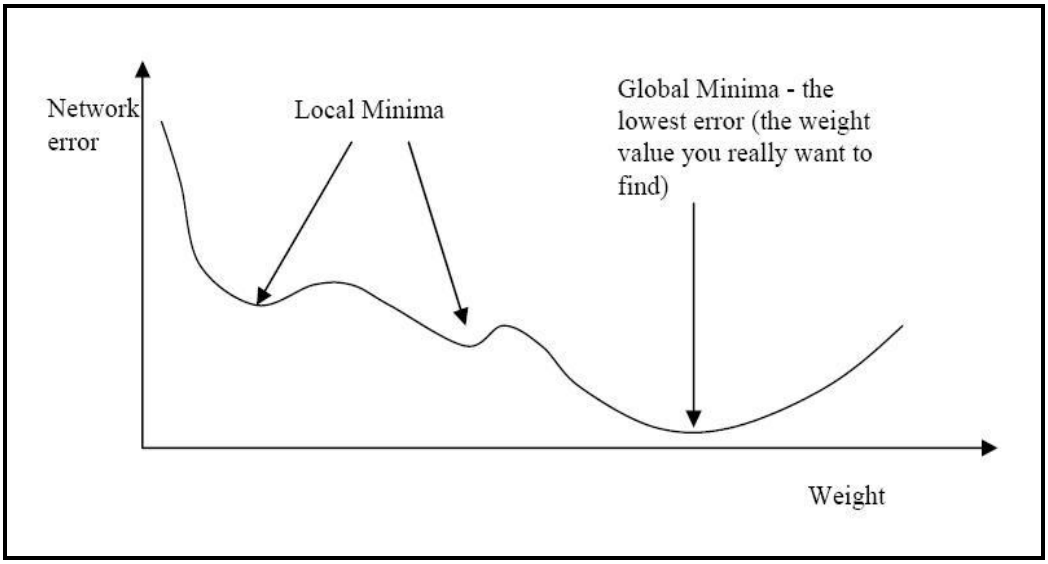


Figure 2.13: An Error Surface

2.1.5.2 The learning Rate

An important consideration in the Back Propagation algorithm is the learning rate $\boldsymbol{\eta}$, which determines by how much we change the weights $\boldsymbol{w}$ at each step. If $\boldsymbol{\eta}$ is too small, the algorithm will take a long time to converge (reaching an optimal set of weights that give the solution) (Hagan et al.,1996). The Influence of choosing a small learning rate on the error surface is illustrated in Figure 2.14.

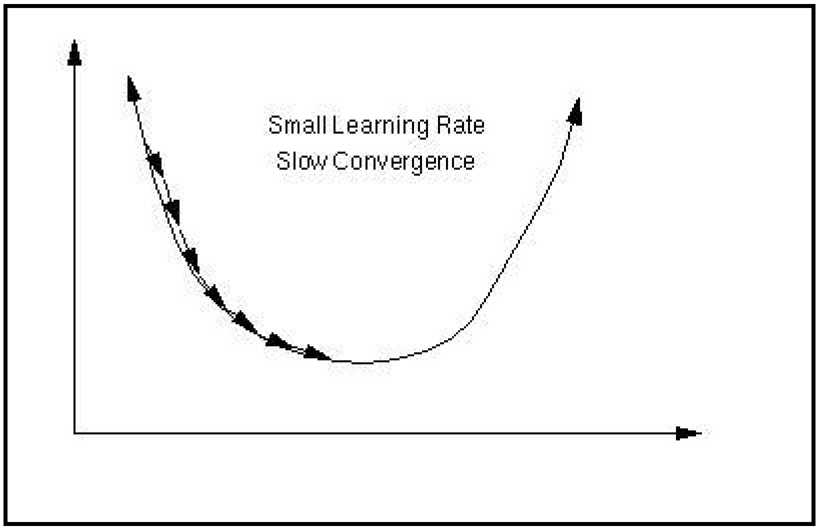


Figure 2.14: The Influence of Choosing a Small Learning Rate on the Error Surface

Conversely, if η is too large, we may end up bouncing around the error surface out of control, which make the algorithm to diverge from the solution. The influence of choosing a large learning rate is illustrated in Figure 2.15.

The correct setting for the learning rate is application-dependent, and is typically chosen by experiment; it may also be time-varying, getting smaller as the algorithm progresses (Hagan et al.,1996).

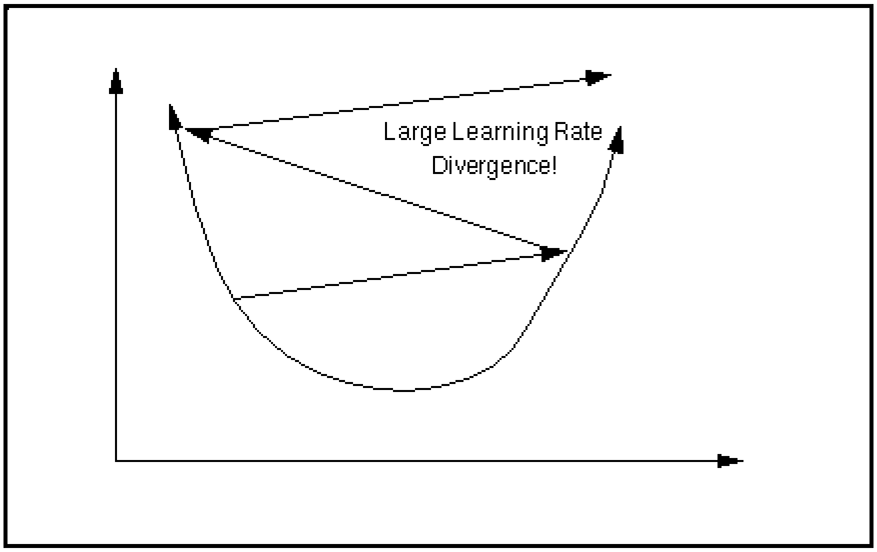


Figure 2.15: The Influence of Choosing a Large Learning Rate on the Error Surface

2.1.5.3 The Back Propagation Steps in Details

The architecture of Backpropagation as shown in **Figure 2.16** work accordingly to the following:

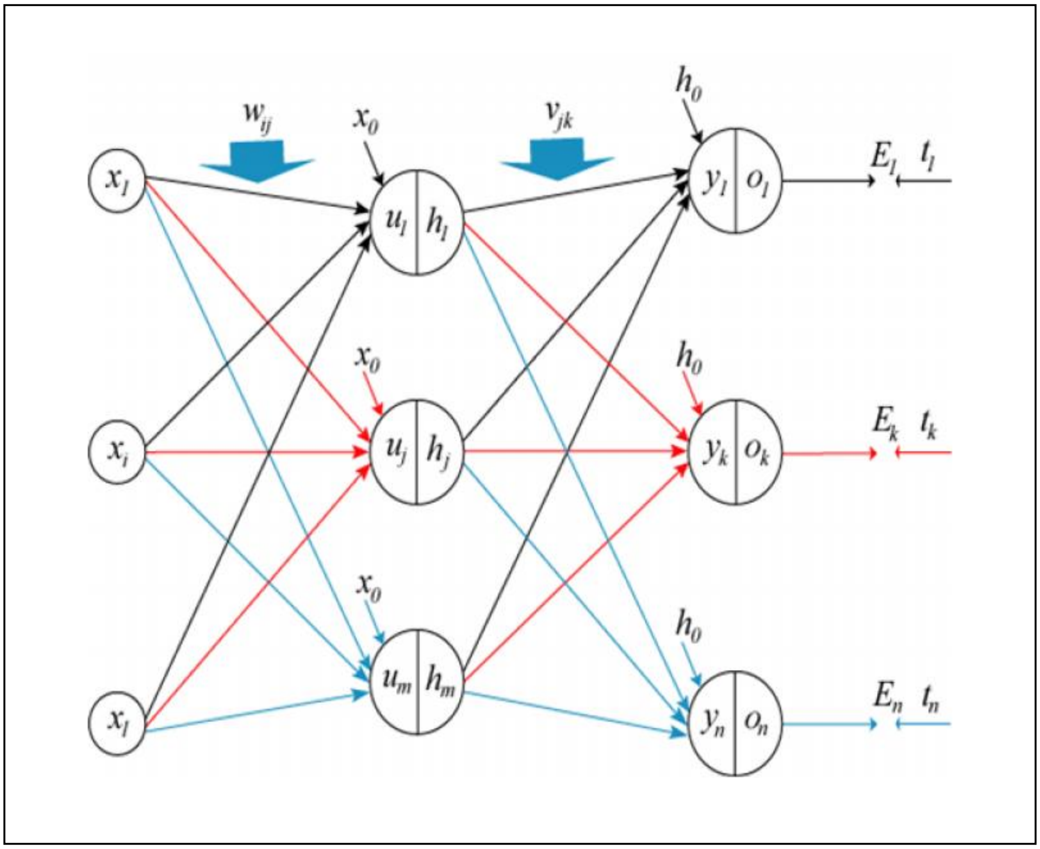


Figure 2.16: The architecture of Backpropagation

Step 1: Initialize the weights of the randomly with numbers between -1 and 1.

$$w_{ij} = random\ ([-1,+1]) \qquad 0 \le i \le l, 1 \le j \le m$$

$$v_{ij} = random\ ([-1,+1]) \qquad 0 \le j \le m, 1 \le k \le n$$

Step 2: Present the pattern p to the network, where p is one pattern from the whole pattern P:

$$p = (x_1, x_2, x_3, x_i, ... x_l) \in R^l \qquad p \in P$$

Step 3: Compute the values of the hidden-layer nodes using the formula:

$$u_j = \sum_{i=0}^{l} w_{ij} \times x_i \qquad , h_j = f(u_j) \qquad ,1 \le j \le m$$

Step 4: Calculate the values of the output nodes using the formula:

$$y_k = \sum_{j=0}^{m} v_{ik} \times h_j \qquad , o_k = f(y_k) \qquad ,1 \le k \le n$$

Step 5: Compute the errors in the output layer using:

$$\delta o_k = (t_k - o_k) o_k (1 - o_k) \qquad ,1 \le k \le n$$

Step 6: Compute the errors in the hidden layer using:

$$\delta h_j = h_j (1 - h_j) \sum_{k=1}^{n} \delta o_k \times v_{jk} \qquad ,1 \le j \le m$$

Step 7: Adjust the weights between the output layer and the hidden according to:

$$v_{jk}^{(new)} = v_{jk}^{(old)} + \eta \times \delta o_k \times h_j \qquad ,0 \le j \le m, 1 \le k \le n$$

Step 8: Adjust the weights between the hidden layer and the input layer according to:

$$w_{ij}^{(new)} = w_{ij}^{(old)} + \eta \times \delta h_j \times x_i \qquad ,0 \le i \le l \qquad ,1 \le j \le m$$

Step 9: Repeat step 2 through 8 for each element of training set, until the Total Mean Squared Error **(TMSE)** ≤ E, E is an error threshold that is used to stop the training, error E defined as the difference between the desired t_k and current o_k output where (Hagan et al.,1996) :

$$TMSE = \frac{1}{n}\sum_{k=1}^{n}(t_k - o_k)^2$$

We notice from step 7 and 8 that these steps are the ones that change the weights of the network based on the following general formula:

$$w_{ij}^{(new)} = w_{ij}^{(old)} + \Delta w_{ij}$$

The term Δw_{ij}, is the amount of change in the current weight, this term in the back propagation algorithm is equal to; $\eta \times \delta o_i \times h_j$ where the term $\delta o_i \times h_j$ represent the partial derivative of the error with respect to the weight ($\frac{\partial E}{\partial w_{ij}}$) (Hagan et al., 1996).

The whole idea behind the partial derivative is; to gradually, but consistently, decrease the output error by adjusting the weights. The trick is to figure out how to adjust the weights. Intuitively, we know that if a change in a weight will increase (decrease) the error, then we want to decrease (increase) that weight. Mathematically, this means that we look at the derivative of the error with respect to the weight: $\frac{\partial E}{\partial w_{ij}}$, once we find this derivative, we will update the weight via the following:

$$\Delta w_{ij} = \eta \times \frac{\partial E}{\partial w_{ij}}$$

2.1.6. Resilient Back Propagation Algorithm (Rprop)

As discussed in subsection 2.1.5.1 a Back Propagation algorithm will experience a local minimum that occurs because the algorithm always changes the weights based on the value of the partial derivative. So in order to solve this problem; the Resilient Back Propagation **(Rprop)** was devised (Riedmiller and Braun, 1993) to eliminate the harmful influence of the magnitude of the partial derivative. As a consequence, only the sign of the derivative is considered to indicate the amount of the weight update. To achieve this, an update-value Δ_{ij} is introduced for each weight $\boldsymbol{w_{ij}}$, which solely determines the size of the weight-update (Riedmiller and Braun, 1993). This adaptive update-value evolves during the learning process**,** according to the following learning-rule:

$$\Delta_{ij}^{(t)} = \begin{cases} \eta^{+} \times \Delta_{ij}^{(t-1)} & , \; if \dfrac{\partial E}{\partial w_{ij}}^{(t-1)} \times \dfrac{\partial E}{\partial w_{ij}}^{(t)} > 0 \\ \eta^{-} \times \Delta_{ij}^{(t-1)} & , \; if \dfrac{\partial E}{\partial w_{ij}}^{(t-1)} \times \dfrac{\partial E}{\partial w_{ij}}^{(t)} < 0 \\ \Delta_{ij}^{(t-1)} & , \; else \end{cases}$$

$$where \;\; 0 < \eta^{-} < 1 < \eta^{+}$$

The adaptation-rule works as follows: every time the partial derivative of the corresponding weight $\boldsymbol{w_{ij}}$ changes its sign, which indicates that the last update was too big and the algorithm has jumped over a local minimum, the update-value Δ_{ij} is decreased by the factor η^{-} (which was found by experiments that the best value for it is equal to **0.5**)**.** If the derivative retains its sign, the update-value is slightly increased by the factor η^{+} (which

was found by experiments that the best value for it is equal to **1.2**) in order to accelerate convergence in shallow regions.

Once the update-value for each weight is adapted, the weight-update itself follows a very simple rule: if the derivative is positive, the weight is decreased by its update-value, if the derivative is negative, the update –value is added:

$$\Delta w_{ij}^{(t)} = \begin{cases} -\Delta_{ij}^{(t)}, & if \ \dfrac{\partial E}{\partial w_{ij}}^{(t)} \rangle \ 0 \\ +\Delta_{ij}^{(t)}, & if \ \dfrac{\partial E}{\partial w_{ij}}^{(t)} \langle \ 0 \\ 0, & else \end{cases}$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)}$$

However, there is one exception: If the partial derivative changes sign, i.e. the previous step was too large and the minimum was missed, the previous weight-update is reverted:

$$\Delta w_{ij}^{(t)} = -\Delta w_{ij}^{(t-1)}, \quad if \ \frac{\partial E}{\partial w_{ij}}^{(t-1)} \times \frac{\partial E}{\partial w_{ij}}^{(t)} < 0$$

Due to that 'backtracking' weight-step, the derivative is supposed to change its sign once again in the following step. In order to avoid a double punishment of the update value,

there should be no adaptation of the update-value in the succeeding step. In practice this can be done by setting $\frac{\partial E}{\partial wij}^{(t-1)} = 0$ in the Δ_{ij} adaptation rule above.

In the Rprop the weights and update-values are changed every time the whole pattern set has been presented once to the network (learning by epoch). Therefore, the term $\frac{\partial E}{\partial wij}^{(t)}$ denotes the summed gradient information over all patterns.

Rprop Algorithm

The following pseudo-code fragment shows the kernel of the **RPROP** adaptation and learning process. The **minimum (maximum)** operator is supposed to deliver minimum (maximum) of two numbers; the **sign** operator returns +1, if the argument is positive, -1, if the argument is negative and 0 otherwise. While $\Delta_{min} = 10^{-6}$ and $\Delta_{max} = 50$.

For all weigths and biases {

$$\textbf{if (} \quad \frac{\partial E}{\partial w_{ij}}^{(t-1)} \times \frac{\partial E}{\partial w_{ij}}^{(t)} \quad > \quad 0) \quad \textbf{then} \quad \{$$

$$\Delta_{ij}^{(t)} = \textbf{minimum} \ (\Delta_{ij}^{(t-1)} \times \eta^{+} \quad , \Delta_{\max})$$

$$\Delta w_{ij}^{(t)} = -\textbf{sign} \ (\frac{\partial E}{\partial w_{ij}}^{(t)}) \times \Delta_{ij}^{(t)}$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)}$$

}

$$\textbf{else} \quad \textbf{if} (\quad \frac{\partial E}{\partial w_{ij}}^{(t-1)} \times \frac{\partial E}{\partial w_{ij}}^{(t)} \quad < \quad 0) \quad \textbf{then} \quad \{$$

$$\Delta_{ij}^{(t)} = \textbf{maximum} \ (\Delta_{ij}^{(t-1)} \times \eta^{-} \quad , \Delta_{\min})$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} - \Delta w_{ij}^{(t-1)}$$

$$\frac{\partial E}{\partial w_{ij}}^{(t)} = 0$$

}

$$\textbf{else} \quad \textbf{if} (\quad \frac{\partial E}{\partial w_{ij}}^{(t-1)} \times \frac{\partial E}{\partial w_{ij}}^{(t)} \quad = \quad 0) \quad \textbf{then} \quad \{$$

$$\Delta w_{ij}^{(t)} = -\textbf{sign} \ (\frac{\partial E}{\partial w_{ij}}^{(t)}) \times \Delta_{ij}^{(t)}$$

$$w_{ij}^{(t+1)} = w_{ij}^{(t)} + \Delta w_{ij}^{(t)}$$

}

}

2.1.7. Designing Issues of Artificial Neural Networks

There are several issues to be considered when designing a neural network application, aiming to achieve a good generalization performance – i.e. to have the outputs of the network approximate well target values given inputs that are not in the training set. These issues will be discussed in the following subsections.

2.1.7.1 Selection of an ANN Topology

When it comes to decide what ANN topology to be used and which training algorithm to apply to this topology, there is no easy answer. Selection of a suitable combination of these two design components is often done experimentally and it is problem dependent (Mehrotra et al., 1996).

Other issues closely related to the network topology include the number of hidden layers and hidden units to be used. In a wide variety of applications a linear model can be sufficient for developing a good classification system. In that case no hidden layers will be needed (Fausett, 2001). However, in applications where a nonlinear model is needed, a MLP with one hidden layer with arbitrary large number of units typically suffices for a good approximation. The optimum number of hidden units depends on the number of input features, the size of the training set, the architecture and the choice of functions and parameters in the algorithm (Veropoulos, 2001). Although a lot of attempts have been made in order to determine a general rules that give the most appropriate number of hidden units to be used for training the network (Mehrotra et al., 1996), none of them can be considered reliable enough or true for any situation.

Another issue is to determine how many training samples are required for successful learning; this is solved in practice by trial and error (Mehrotra et al., 1996). These issues are complex because there is considerable dependence on the specific problem being attacked using a neural network (Veropoulos, 2001).

2.1.7.2 Pre-Processing and Post-Processing

The input data should be in such a form that a neural network can process it, this means numeric values. Furthermore, the numeric input values may be pre-processed in a way that the network's learning task is easier. In many practical applications, the choice of the pre-processing method will be one of the most significant factors in determining the performance of the ANN (Hagan et al., 1996). Pre-processing is always performed before the training process begins. Sometimes the output data are also processed. This is called post-processing which is converting back the output into the same units that were used in the original dataset.

One commonly used pre-processing method is data standardization; standardizing a vector most often means subtracting a measure of location and dividing by a measure of scale. For example, you might subtract the mean and divide by the standard deviation, thereby obtaining a "standard normal" random variable with mean 0 and standard deviation 1.

The average (mean) is a measure of the middle value of the data set. If n numbers are given, each number denoted by $\boldsymbol{a_i}$, where i = 1,..,n, the mean is the [sum] of the $\boldsymbol{a_i}$'s divided by n or:

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{N} a_i$$

While the standard deviation shows how much variation there is from the "average" (mean). A low standard deviation indicates that the data points tend to be very close to the

mean, whereas high standard deviation indicates that the data are spread out over a large range of values (Ghahramani, 2000). The standard deviation S_N is defined as follows:

$$S_N = \sqrt{\frac{1}{n}\sum_{i=1}^{N}(x - \overline{x})^2}$$

In order to obtain a dataset with zero mean and unit standard deviation, we use the following :

X_i = Value of the raw input variable X for the i^{th} training case.

S_i = Standardized value corresponding to X_i.

N = Number of training cases.

Standardize X to mean 0 and standard deviation 1 by:

$S_i =$ (Xi – mean(X)) / Standard Deviation(X).

Note that statistics such as the mean and standard deviation are computed from the training data, not from the validation or test data. The validation and test data must be standardized using the statistics computed from the training data.

2.1.7.3 Generalization Performance of an ANN

One of the major advantages of neural networks is their ability to generalize; this means that a trained network could classify data that it has never seen before from the same class as the learning data. In real world applications developers normally have only a small part of all possible patterns for the generation of a neural network (Hagan et al.,1996). To reach the best generalization, the dataset should be split into three parts:

- **The training set** that is used to train a neural network. The error of this dataset is minimized during training.

- **The validation set** that is used to determine the performance of a neural network on patterns that are not trained during learning.
- **A test set** for finally checking the over all performance of a neural network.

The learning should be stopped in the minimum of the validation set error. At this point the network generalizes best. When learning is not stopped, overfitting occurs and the performance of the network on the whole data decreases, despite the fact that the error on the training data still gets smaller. After finishing the learning phase, the network should be finally checked with the test set (Veropoulos,2001). An example of an overfitted network is illustrated in Figure 2.17

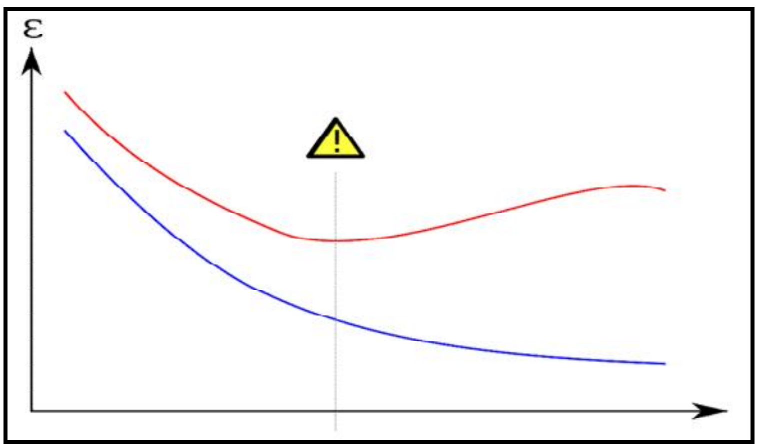


Figure 2.17: Overfitting in Neural Networks, Training error is shown down and validation error is shown up

This technique is called early stopping and it is used for improving the generalization of a neural network during training and preventing it from overfitting (Haykin 2001). Where the basic concept is that the performance of the ANN against the training set will continue to improve with more training, and this is also true for its performance against the validation set (Machado1996). However, if the ANN starts

overfitting the training set, its performance then against the validation set will start to decrease. Therefore, the validation error will increase; this is an indicative of a poor generalization and can be used for stopping the classifier from further training (Haykin 2001). The weights giving best performance against the validation set can then be accepted as the set of weights that gives the ANN best generalization ability. Since the error surface is not always smooth, it is not certain at this point of the training if the validation error is the global minimum (Hagan et al.,1996). So, it is common practice to continue training for several iterations (based on the judgment of the designer) in case the ANN performance against the validation set improves again. At the same time the configuration and set of parameters of the ANN that gave best results until this point are always stored (Veropoulos,2001).

2.2 Using Artificial Neural Networks in the Medical Problems

Neural networks, with their remarkable ability to derive meaning from complicated or imprecise data, can be used to extract patterns that are too complex to be noticed by either humans or other computer techniques, therefore, due to their capabilities they have been used widely in the medical field. Section 2.2.1 discusses some of the applications of neural networks in the medical domain in general, where section 2.2.2 presents some of the applications of neural networks in the cardiology domain.

2.2.1. Applications of Neural Networks in the Medical Domain

Previous studies sought ways of capitalizing the use of neural networks in the diagnosis of breast cancer to improve the results of breast cancer screenings by evaluating the performance of ANNs to predict an outcome (cancer/not cancer). Paulin and Santhakumaran (2011) presented a study on the classification of breast cancer using feed forward ANN, the performance of the network is evaluated using Wisconsin Breast Cancer Dataset **(WBCD)** where an accuracy of 99.28% using the Levenberg-Marquardt **(LM)** training algorithm was obtained. Also, (Kiyan and Yildirim, 2003) evaluated the performance of the Radial Basis Network **(RBF)**, General Regression Neural Network **(GRNN)**, MLP and Probabilistic Neural Network **(PNN)** on the WBCD dataset and they concluded that the most suitable neural network model for classifying the WBCD is the GRNN with accuracy 98.8% while the classification performances for the other types of neural networks were 96.18% for RBF, 97.0% for PNN, and 95.74% for MLP.

Several studies proved that neural networks are useful for the differentiation of two forms of chronic active hepatitis; Nakano et al., (1996) investigated the ability of neural

networks to discriminate mild and severe chronic active hepatitis, where the result showed a network capable of correctly predicting the diagnosis for 78% of the cross-validation group. Bascil and Temurtas, (2009) presented another study for hepatitis diagnosis using a multilayer neural network as a machine learning technique and LM algorithm as the training algorithm, the architecture was applied on the dataset that was obtained from the University of California Irvine **(UCI)** machine learning repository, the study obtained a classification accuracy of 91.87% via ten-fold cross validation. On the other hand there were some studies that showed the ability of neural networks in predicting diseases complication; Vahdani et al. (2009) considered in their study on predicting the presence or absence of cirrhosis in patients with chronic hepatitis B by using routine laboratory findings; they concluded that in patients with chronic hepatitis B, if the ANN value of certain laboratory manifestations is negative, there will be a 95% chance of having a negative liver biopsy.

In the field of diabetes diagnosis; (Kayaer and Yildirim, 2003) applied two different neural network structures, which are MLP and GRNN to the Pima Indians Diabetes **(PID)** medical data obtained from the UCI machine learning repository. They achieved 80.21% classification accuracy using GRNN and they have also reported 77.08% classification accuracy using multilayer neural network with LM training algorithm. Furthermore (Temurtas et al., 2009) in their study realized a PID diagnosis system; for this purpose, a MLP neural network with LM training algorithm was used, the study achieved 79.62 % accuracy using a ten-fold cross validation approach and it achieved 82.37% classification accuracy using the conventional validation approach (one training set and one test set).

AlTimemy and Al Naima (2010) in their study presented the prediction of kidney dysfunction using different neural network approaches; PNN, SOM and MLP neural

network, where the MLP was trained with back propagation algorithm. These networks were used as classifiers to predict whether the kidney was normal or it would have a dysfunction. The networks were trained by a dataset collected from one of the private clinical laboratories in Baghdad. In this study the PNN gave more accurate prediction of kidney dysfunction with accuracy 99% while the accuracy of SOM was 98% and the accuracy for MLP was 95%.

2.2.2. Applications of Neural Networks in the Cardiology Domain

In the cardiology domain, multiple studies showed the effectiveness of applying ANN in building clinical decision support systems for assisting physicians in their decisions.

Concerned with the diagnosis of congestive heart failure and chronic obstructive pulmonary disease (since they have similar symptoms which can make their distinction difficult especially at the time of admission or where the access to echocardiography is limited) (Mehrabi et al., 2009) used a MLP and RBF neural networks to differentiate between patients suffering from one of these diseases, the MLP led to an accuracy of 84.95% where RBF network resulted an accuracy of 85.1%.

Yan et al. (2006) presented a multilayer perceptron based decision support system to support the diagnosis of heart diseases; the system was applied to five different heart diseases categories using an updated Back Propagation training algorithm. The results showed that the proposed system achieved an accuracy of 90% proving its usefulness in the support of clinical diagnosis decisions for heart diseases.

Bakul et al. (2009) presented a study that obtained an accuracy of 81.82% using a neural network with Back Propagation algorithm for the diagnosis of coronary artery disease using a dataset that contained features that were extracted from a 12-lead ECG.

Stefeko (2007) confirmed that it is possible to localize the narrowing in a particular coronary artery of four main coronary vessels with the aid of a MLP Backpropagation neural network applied for traditional ECG exercise test interpretations, where the study obtained an accuracy of 96%.

Turkoglu et al. (2002) presented an expert diagnosis system for the interpretation of the Doppler signals of the heart valve diseases using a back-propagation neural network. The test results showed that this system was effective to detect Doppler heart sounds. The correct classification rate was about 94% for normal subjects and 95.9% for abnormal subjects.

Chapter Three

Data Preparation and System Architecture

3. Data Preparation and System Architecture

Since ANNs perform quit well with complex tasks such as classification problems using abstract data, they have been applied to many areas in the medical field for the analysis of various diseases and conditions. Application of ANNs to the diagnosis for precise types of heart diseases is a fairly new area, and not much analysis has been performed with ANNs. In this chapter section 3.1 illustrates how data was collected and prepared in order to build the heart disease CDSS. Section 3.2 concerns about the system architecture for the heart disease CDSS.

3.1. Data Preparation

When a patient reports to a physician, a large number of possibly relevant inputs must be considered during the diagnosis. A physician reaches an accurate diagnosis or treatment decision based upon observations, the patient's answers to questions and physical examinations or lab results.

In this thesis, the heart disease dataset used for testing and training the system consisted of a total of 162 cases (in which each disease has a total of 54 cases) gathered from the cardiology clinics at JUH, located in Amman, Jordan. Moreover, 58 variables essential to the diagnosis of the heart diseases of interest were interpreted from each patient's medical record. These variables can be divided into six categories:

i. Basic information of a patient (including the age and the gender).

ii. The patient's history (10 factors in total).

iii. Symptoms (16 factors in total).

iv. Physical examination (10 factors in total).

v. Blood lab results (7 factors in total).

vi. ECG findings (12 factors in total).

In Figure 3.1 the datasheet that was used in gathering the patent's variables is illustrated. We notice from the figure that most of the attributes were assigned to have a yes or no value to indicate the presence or absence of an attribute, some attributes if were presented the type of them must be determined either as a severe or mild. Also, the following attributes will be assigned values as follows:

- Smoking (that belongs to the patient's history category), one of three values will be assigned to; a smoker or not a smoker or an ex-smoker,
- The heart beat (that belongs to the physical examination category); a regular or irregular will be assigned to this attribute.
- First and second heart sounds (that belong to the physical examination category); Normal or abnormal will be assigned to these attributes. While the third and fourth heart sounds are either presented or not.
- Jugular venous pressure (that belongs to the physical examination category); raised or not will be assigned to this attribute.
- One value was assigned for the gender; either female or male.
- The Age, diastolic blood pressure, systolic blood pressure, pulse and all blood lab results were assigned the actual values the patient has for these attributes.

Patient's Datasheet

Diseases Type

i. AF
ii. MI
iii. USA

i. Patient's Basic information

Age:_________
Gender: Female/Male

ii. The patient's history

- Upper Respiratory Infection: Yes/No
- Chronic Pulmonary diseases History : Yes/No
- Family History of Hypertension: Yes/No
- Family History of Diabetes : Yes/No
- Family History of Heart Disease: Yes/No
- Patient's Previous Heart Disease History: Yes/No
- Dyslipidemia: Yes/No
- Diabetes: Yes/No
- Hypertension: Yes/No
- Smoking: Smoker/Not a Smoker/Ex-smoker

iii. Symptoms

- Chest Pain or chest discomfort: Yes(Severe/Mild) /No
- Chest pain referred to left arm/shoulder : Yes/No
- Chest pain referred to right arm/shoulder: Yes/No
- Symptoms appeared with exertion: Yes/No
- Symptoms appeared at rest: Yes/No
- Shortness of Breath: Yes(Severe/Mild)/No
- Epigastric Pain: Yes(Severe/Mild)/No
- Palpitations: Yes/No
- Lower Limb Edema: Yes/No
- Nausea: Yes(Severe/Mild)/No
- Vomiting: Yes(Severe/Mild)/No
- Sweating: Yes(Severe/Mild)/No
- Dizziness: Yes(Severe/Mild)/No
- Headache: Yes(Severe/Mild)/No
- Fatigue: Yes(Severe/Mild)/No
- Syncope: Yes/No
- Cough: Yes(Severe/Mild)/No

iv. Physical examination

- Diastolic/Systolic Blood Pressure: ___/___
- Pulse: ________
- Heart Beat: Regular/Irregular
- Heart Murmur: Yes/No
- Jugular venous pressure: Raised/Not Raised
- First Heart Sound: Normal/Abnormal
- Second Heart Sound: Normal/Abnormal
- Third Heart Sound: Presented/Not presented
- Fourth Heart Sound: Presented/Not presented

v. Blood Lab Tests

- Tropinin: _______
- CK: ____
- CK-MB: _____
- AST: _____
- ALT:_____
- HB: _____
- WBC: _____

vi. ECG Findings

- T-wave inverted: Yes(Severe/Mild)/No
- Big T-wave: Yes(Severe/Mild)/No
- ST-Elevation: Yes(Severe/Mild)/No
- ST-Depression: Yes(Severe/Mild)/No
- Q-Wave: Yes(Severe/Mild)/No
- RBBB: Yes(Severe/Mild)/No
- LBBB: Yes(Severe/Mild)/No
- Right Axis Deviation: Yes(Severe/Mild)/No
- Left Axis Deviation: Yes(Severe/Mild)/No
- Left Ventricular hypertrophy:Yes(Severe/Mild)/No
- Right Ventricular hypertrophy:Yes(Severe/Mild)/No
- Tachyarrhythmias: Yes(Severe/Mild)/No

Figure 3.1: Patient's Datasheet

3.2. System Architecture

The following subsections explain how the input variables that were used in the training and testing processes were encoded for building the heart disease CDSS, and how the number of hidden layers and hidden neurons in each layer were chosen.

3.2.1 Input Variables Encoding Scheme

All neural networks receive numerical inputs; therefore, the 58 variables were encoded into numerical values using the following encoding scheme:

i. Numerical variables such as age, systolic and diastolic blood pressure, pulse and all lab results were reserved the same with no changes since they are already in a numerical format.

ii. Variables with two independent attributes, such as the gender, all attributes from the patient's history except for the smoking attribute, some symptoms and some physical examination attributes are encoded with binary values (0, 1). For instance for the gender attribute; 0 represents a female and 1 represents a male; Also, 1 is adopted for the mentioned attributes to indicate the presence of the attribute in that category, and 0 for the absence of it.

iii. Variables with two independent attributes such as some symptoms and all the ECG findings must determine the severity of them if they were presented, so they will be encoded by three values (0, 0.5, 1) where 0 presents the absence of the attribute, 0.5 presents a mild presence of the attribute and 1 presents a severe presence of the attribute.

iv. Variables with three independent attributes, such as smoking are encoded using the ternary values (0, 0.5, 1); with 0 representing that the patient is not a smoker, 0.5 representing an ex-smoker patient and 1 a smoker patient.

After encoding all variables, the training dataset was standardized to have a zero mean and a unit standard deviation as explained in subsection 2.1.7.2, and based on the information that was collected form the training dataset during the standardization; the validation and test datasets were also standardized to have a zero mean and a unit standard deviation.

3.2.2. Number of Hidden Layers and Hidden Neurons

Optimizing the number of hidden layer and neurons in each hidden layer for a feed forward neural network remains one of the unsolved tasks in this research area. Thus several researchers have proposed some general rules for determining an optimal number of hidden neurons for any application. One of the rules is: one hidden layer is enough, and to find the number of neurons for the first hidden layer; add the number of neurons in the input layer to the number of neurons in the output layer and divide the result by two. Another known rule is: if more than one hidden layer must be used to achieve a less generalization error; then the number of neurons in each hidden layer except the first one will be the number of neurons in the previous layer divided by two (Xu and Chen, 2008). Moreover, Kavzoglu as cited in (Yan et al., 2006) has shown that in most situations, there is no way to determine the best number of hidden layers without training several networks and estimating the generalization error of each, because the best number of hidden layers

and units depends mainly in a complex way on: the number of input and output units, the number of training cases and the complexity of the classification problem to be learned (Xu and Chen, 2008).

Based on this, we used an iterative process to determine the best number of hidden layers and neurons in each hidden layer. In the iterative process a ten-fold cross validation technique (more details will be provided in chapter 4) is used to access the generalization for each architecture. The whole process works as follow:

Step1. Start testing the network architecture with only one hidden layer; the following equation will be applied to find the number of neurons in the first hidden layer:

$$n_f = (n_i + n_o) / 2$$

Where n_f is the number of neurons in the first hidden layer, n_i is the number of neurons in the input layer and n_o is the number of the neurons in the output layer. If n_f was not a fixed number then apply the ceil and the floor operations, so you will get two values, and for more precise results take another number which is floor(n_f)-1, so you will have three numbers of neurons in the first layer to start with.

Step2. Add another layer; the number of neurons in this layer will be half the number of neurons in the previous layer.

Step3. Repeat step two until the number of the hidden neurons in the layer is one.

Chapter Four

Experimental Results and Evaluation

4. Experimental Results and Evaluation

In this chapter, Section 4.1 introduces the evaluation method that was used to assess the generalization of the ANN models. Section 4.2 presents multiple ANN architectures that were built in order to choose the best ANN model for building the CDSS for the heart diseases diagnosis. Finally Section 4.3 introduces the detailed results for the best ANN model that was obtained in Section 4.2.

4.1. Cross Validation

Cross validation is a technique used for estimating the performance of a predictive model, also it can be referred as estimate of accuracy, sometimes referred as rotation estimation, and it is determined by the overall number of correct classifications divided by the total number of instances in the dataset (Yan et al., 2006). One round of cross-validation involves partitioning a sample of data into subsets, performing the analysis on one subset (called the training set), and testing the analysis on the other subset (called the testing set). As we discussed before in subsection 2.1.7.3 ANN may go through an overfitting. Therefore, a third subset called the validation set is used to avoid the overfitting of a network. Moreover, to reduce variability and to avoid bad splits that may produce overfitting, multiple rounds of cross-validation are performed using different partitions, and the tested results are averaged over the rounds.

In this thesis a ten-fold cross validation was used to access the generalization of all network models that were built. A percentage of 80%, 10% and 10% data split of the complete data samples was applied to represent the training, validation and test subsets respectively (Yan et al., 2006), this provides for each fold of the ten folds a total of 132, 15

and 15 sample cases for the train, validation and test subsets respectively. In each fold each disease in the training set will be represented by 44 samples of the total samples of the training set. Also, each disease in the validation set will be represented with 5 samples and 5 samples for the test set.

In order to find the classification accuracy for a model, we must first calculate the classification accuracy for each individual fold and after that averaging the results over the ten folds as follows:

- Compute the classification accuracy for each fold; the classification accuracy A_{fi} of an individual fold f_i depends on the number of samples correctly classified and is evaluated by the formula:

$$A_{fi} = (t/n) * 100\%$$

Where t is the number of sample cases correctly classified, and n is the total number of sample cases.

- Find the average classification accuracy for the ten folds as follows:

$$Average\ Classification\ Accuracy = \frac{1}{10}\sum_{i=1}^{10} A_{f_i}$$

4.2. Models' Building

In order to investigate the best system architecture for building the CDSS for the heart diseases diagnosis; multiple system architectures were built using the iterative process that was explained in subsection 3.2.2. The different architectures were compared based on the classification accuracy each architecture has obtained. The model with the highest

classification accuracy (less generalization error) was chosen as the final model for building the CDSS.

The different ANN architectures have been built and tested using MATLAB® version 2010a, which was run under windows Vista Home Premium SP1 OS with an AMD Turion X2 Dual-Core processor of 2.00 GHz, and 3.00 GB of RAM.

In this thesis for all models:

- A Feedforward Backpropagation neural network was used for building all models.
- The number of neurons in the input layer was 58 (representing number of features for each sample).
- The number of neurons in the output layer was 1 (representing one class of the diseases the neural will generate).
- The training algorithm that was used for training the models was the Rprop.
- LF and TANH were the activation functions that were used in the output layer for each architecture, and the results obtained using both functions were compared.
- The activation function that was used for all the hidden neurons in all hidden layers was TANH.
- The following values were used 0.08, 0.0000001, and 6 for the initial delta, the performance goal error and the number of validation checks to avoid the overfitting of the network respectively.

4.2.1. Models' Building using One Hidden Layer

Since the input layer contains 58 neurons and the output layer contains 1 neuron, the number of neurons in the first hidden layer will be 29.5 based on the equation in step 1 from the iterative process in subsection 3.2.2. Because the result is not a fix number, applying the ceil operation will produce 30, applying the floor operation will produce 29, therefore, the network can be trained with two different numbers of neurons, and to make the testing more precise another number is used, which is 28. Accordingly the network will be tested with three numbers of neurons; 28, 29 and 30.

Each number will be tested twice, once with a LF and another with a TANH as an activation functions for the output layer. Table 4.1 illustrates the results of 6 different architectures, the best architecture that produced the highest classification accuracy is (58-28-1) with an TANH activation function in the output layer; this architecture represents 58 input variables, one hidden layer of 28 neurons and one output layer, the classification accuracy for this architecture is 72.6%. All the remaining architectures produced low classification accuracy varying in an interval [50%-70.6%].

Table 4.1: Classification Accuracy with one hidden layer

Network Architecture	Activation Function	Classification Accuracy
58-28-1	TANH	72.6%
58-28-1	LF	50 %
58-29- 1	TANH	70.6%
58-29- 1	LF	53.3%
58-30-1	TANH	65.3%
58-30-1	LF	56%

4.2.2. Models' Building using Two Hidden Layers

For the first hidden layer we obtained three different numbers of neurons; 28, 29 and 30, so by applying step 2 from the iterative process in subsection 3.2.2, number of neurons in the second layer will be as follows:

i. When the number of neurons in the first hidden layer is 28, the number of neurons in the second hidden layer will be 28/2 which is 14 neurons. The network architecture will be 58-28-14-1.

ii. When the number of neurons in the first hidden layer is 29, the number of neurons will be 29/2 which is 14.5, since this is not a fix number applying the floor operation will produce 14, applying the ceil operation will produce 15; therefore, this will produce two architectures 58-29-14-1, and 58-29-15-1.

iii. When the number of neurons in the first layer is 30 the number of neurons in the second layer will be 15. The network architecture will be 58-30-15-1.

Consequently, four different architectures will be tested when the number of the hidden layers is two. As noticed from Table 4.2, the best classification accuracy is 84.6% when the network architecture is 58-28-14-1 with TANH. The results for the other architectures vary in an interval [64% - 74.6%].

Table 4.2: Classification Accuracy with two hidden layers

Network Architecture	Activation Function	Classification Accuracy
58-28-14-1	TANH	84.6%
58-28-14-1	LF	64%
58-29-14-1	TANH	74.6%
58-29-14-1	LF	70%
58-29-15-1	TANH	72%
58-29-15-1	LF	64%
58-30 15-1	TANH	74.6%
58-30 15-1	LF	72%

4.2.3. Models' Building using three Hidden Layers

We obtained four different architectures when the number of hidden layers was two; 58-28-14-1, 58-29-14-1, 58-29-15-1 and 58-30-15-1, when a third hidden layer is added number of neurons in this layer will be as follow:

i. When the network architecture is 58-28-14-1, the number of neurons in the second hidden layer will be 14, therefore, number of neurons in the third hidden layer is 14/2=7. The network architecture will be in this case 58-28-14-7-1.

ii. When the network architecture is 58-29-14-1, the number of neurons in the second hidden layer will be 14, therefore, number of neurons in the third hidden layer is 14/2=7. The network architecture will be in this case 58-29-14-7-1.

iii. When the network architecture is 58-29-15-1, the number of neurons in the second hidden layer will be 15, therefore, number of neurons in the third hidden layer is 15/2=7.5, and since the result is not a fix number applying the floor operation will produce 7, applying the ceil operation will produce 8, then the result is two architectures, 58-29-15-7-1 and 58-29-15-8-1.

iv. When the network architecture is 58-30-15-1, the number of neurons in the second hidden layer will be 15, therefore, number of neurons in the third hidden layer is 15/2=7.5, and since the result is not a fix number applying the floor operation will produce 7, applying the ceil operation will produce 8, then the result is two architectures, 58-30-15-7-1 and 58-30-15-8-1.

Table 4.3 illustrates the classification accuracy when using three hidden layers; the highest accuracy was achieved with the network architecture of the form 58-29-14-7-1 with LF is 89.3 % classification accuracy. The other architectures achieved results varying in an interval [75.3%-86%].

Table 4.3: Classification Accuracy with three hidden layers

Network Architecture	Activation Function	Classification Accuracy
58-28-14-7-1	TANH	75.3%
58-28-14-7-1	LF	78%
58-29-14-7-1	TANH	83.3%
58-29-14-7-1	LF	89.3%
58-29-15-7-1	TANH	86%
58-29-15-7-1	LF	76%
58-29-15-8-1	TANH	79.3%
58-29-15-8-1	LF	82%
58-30-15-7-1	TANH	78%
58-30-15-7-1	LF	84.6%
58-30-15-8-1	TANH	76.6%
58-30-15-8-1	LF	79.3%

4.2.3. Models' Building using Four, Five and Six Hidden Layers

Building the network with four hidden layers needs to use the architectures that were obtained using three hidden layers. Also, to build the network with five hidden layers, the networks that were built with four hidden layers must be used. Finally, the highest number of hidden layer that can be achieved is six, by using the obtained networks with five hidden layers.

Table 4.4 illustrates the results obtained using four hidden layers, we can notice that the highest classification accuracy which was 85.3% was obtained with network architecture 58-29-14-7-4-1 the architectures were built with TANH. The remaining architectures gave results varying in an interval [70.6%-84.6%].

Table 4.4: Classification Accuracy with four hidden layers

Network Architecture	Activation Function	Classification Accuracy
58-28-14-7-3-1	TANH	72%
58-28-14-7-3-1	LF	80.6%
58-28-14-7-4-1	TANH	84.6%
58-28-14-7-4-1	LF	77.3%
58-29-14-7-3-1	TANH	83.3%
58-29-14-7-3-1	LF	83.3%
58-29-14-7-4-1	TANH	85.3%
58-29-14-7-4-1	LF	78.6%
58-29-15-7-3-1	TANH	84.6%
58-29-15-7-3-1	LF	82.6%
58-29-15-7-4-1	TANH	79.3%
58-29-15-7-4-1	LF	70.6%
58-29-15-8-4-1	TANH	72%
58-29-15-8-4-1	LF	79.3%
58-30-15-7-3-1	TANH	82.6%
58-30-15-7-3-1	LF	79.3%
58-30-15-7-4-1	TANH	78.6%
58-30-15-7-4-1	LF	74.6%
58-30-15-8-4-1	TANH	80.6%
58-30-15-8-4-1	LF	76.6%

Table 4.5 illustrates the results obtained using five hidden layers, we can notice that the highest classification accuracy is 84.6% which was obtained using 58-29-15-8-4-2-1 network architecture with TANH. The remaining architectures gave results varying in an interval [53.3%-84%].

Table 4.5: Classification Accuracy with five hidden layers

Network Architecture	Activation Function	Classification Accuracy
58-28-14-7-3-1-1	TANH	84%
58-28-14-7-3-1-1	LF	80.6%
58-28-14-7-3-2-1	TANH	78%
58-28-14-7-3-2-1	LF	78%
58-28-14-7-4-2-1	TANH	72%
58-28-14-7-4-2-1	LF	80%
58-29-14-7-3-1-1	TANH	81.3%
58-29-14-7-3-1-1	LF	82.3%
58-29-14-7-3-2-1	TANH	78%
58-29-14-7-3-2-1	LF	69.3%
58-29-14-7-4-2-1	TANH	78.6%
58-29-14-7-4-2-1	LF	78.6%
58-29-15-7-3-1-1	TANH	80.6%
58-29-15-7-3-1-1	LF	76%
58-29-15-7-3-2-1	TANH	67.3%
58-29-15-7-3-2-1	LF	53.3%
58-29-15-7-4-2-1	TANH	76%
58-29-15-7-4-2-1	LF	80.6%
58-29-15-8-4 -2-1	TANH	84.6%
58-29-15-8-4-2-1	LF	79.3%
58-30-15-7-3-1-1	TANH	77.3%
58-30-15-7-3-1-1	LF	81.3%
58-30-15-7-3-2-1	TANH	74.6%
58-30-15-7-3-2-1	LF	74.6%
58-30-15-7-4-2-1	TANH	75.3%
58-30-15-7-4-2-1	LF	81.3%
58-30-15-8-4-2-1	TANH	70%
58-30-15-8-4-2-1	LF	81.3%

Table 4.6 illustrates the results obtained using six hidden layers, we can notice that the highest classification accuracy is 82.6% which was obtained using 58-29-14-7-4-2-1-1 network architecture with TANH. The remaining architectures gave results varying in an interval [63.3%-82%].

Table 4.6: Classification Accuracy with six hidden layers

Network Architecture	Activation Function	Classification Accuracy
58-28-14-7-3-1-1-1	TANH	77.3%
58-28-14-7-3-1-1-1	LF	80%
58-28-14-7-3-2-1-1	TANH	81.3%
58-28-14-7-3-2-1-1	LF	63.3%
58-28-14-7-4-2-1-1	TANH	74%
58-28-14-7-4-2-1-1	LF	75.3%
58-29-14-7-3-1-1-1	TANH	80.6%
58-29-14-7-3-1-1-1	LF	66.6%
58-29-14-7-3-2-1-1	TANH	68.6%
58-29-14-7-3-2-1-1	LF	65.3%
58-29-14-7-4-2-1-1	TANH	82.6%
58-29-14-7-4-2-1-1	LF	76%
58-29-15-7-3-1-1-1	TANH	82%
58-29-15-7-3-1-1-1	LF	72.6%
58-29-15-7-3-2-1-1	TANH	79.3%
58-29-15-7-3-2-1-1	LF	76%
58-29-15-7-4-2-1-1	TANH	79.3%
58-29-15-7-4-2-1-1	LF	77.3%
58-29-15-8-4-2-1-1	TANH	80%
58-29-15-8-4-2-1-1	LF	76%
58-30-15-7-3-1-1-1	TANH	78.6%
58-30-15-7-3-1-1-1	LF	60.6%
58-30-15-7-3-2-1-1	TANH	80%
58-30-15-7-3-2-1-1	LF	78.6%
58-30-15-7-4-2-1-1	TANH	78.6%
58-30-15-7-4-2-1-1	LF	70%
58-30-15-8-4-2-1-1	TANH	76.6%
58-30-15-8-4-2-1-1	LF	80%

4.2.5. Summary of Models' Building

Table 4.7 illustrates the summary of the best results we obtained using different hidden layers for building the neural network that will be used as the final model for the clinical decision support system.

We started building the models with one hidden layer and ended with 6 hidden layers. Where the best architecture was obtained by using three hidden layers; the architecture is 58-29-14-7-1 and obtained 89.3% classification accuracy. Using one hidden layer did not achieve good results, when another hidden layer has been added the best classification accuracy has risen from 72% to 84.6%. By adding a third layer the highest accuracy has been obtained with a percentage of 89.3%. Adding four, five and six hidden layers lowered the accuracy and did not improve the results.

These results can be explained as follows: if you have too few hidden layers, you will get high training error and high generalization error due to underfitting (Xu and Chen, 2008); this was noticed in the low accuracy we obtained when the network was built with one hidden layer and two hidden layers. Also, if you have too many hidden layers, you may get low training error but still have high generalization error due to overfitting (Xu and Chen, 2008); this was noticed when we built the networks with 4, 5 and 6 hidden layers.

From this we can conclude that experiments are the only way to determine the best neural network architecture that is suitable for solving a specific problem.

Table 4.7: Summary of Results Regarding Multiple Hidden Layers

Number of hidden layers	Best Classification Accuracy %	Best Network Architecture
1	72.60%	58-28-1
2	84.60%	58-28-14-1
3	89.30%	58-29-14-7-1
4	85.30%	58-29-14-7-4-1
5	84.60%	58-29-15-8-4-2-1
6	82.60%	58-29-14-7-4-2-1-1

4.3. Detailed Results of the Best ANN Model and Discussion

The effectiveness of the best neural network architecture which is (58-29-14-7-1) with LF activation function for the output layer was evaluated on the heart disease test set for each fold independently. After that, the average accuracy is computed to determine how well the chosen architecture will generalize for new samples that it never seen before. The following subsections will introduce the results for each test set of the 10 folds (runs), where the experimental results of the ten test sets have also been presented as a confusion matrix; a confusion matrix contains information about actual and predicted classifications done by a classification system. Performance of such a system is commonly evaluated using the data in the matrix. In a confusion matrix, columns represent the predicted data, while rows represent the actual data (Yan et al., 2006).

4.3.1. Results of the First Run

Table 4.8 illustrates the confusion matrix of the first run. The accuracy for the USA disease is ((4/5)*100%) which is 80%, for the MI disease it is ((4/5) * 100%) which is 80%, and for the AF disease it is ((5/5) *100%) is 100%. This gives a total accuracy for the first run by applying the equation in section 4.1; (A_{f1} = (13/15)* 100%) = 86.6%. We can notice from the confusion matrix that 20% from the samples of the USA disease were misclassified, and 20% from the samples of the MI were also misclassified.

Table 4.8: Confusion Matrix of the First Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	4	1	0	80%
MI	1	4	0	80%
AF	0	0	5	100%
Total Accuracy 86.6%				

4.3.2. Results of the Second Run

Table 4.9 illustrates the confusion matrix of the second run. The accuracy for the USA disease is ((5/5)*100%) which is 100%, for the MI disease it is ((5/5) * 100%) which is 100%, and for the AF disease it is ((5/5) *100%) which is 100% .This gives a total accuracy of (A_{f2} = (15/15)* 100%) = 100%. In this run no samples were misclassified.

Table 4.9: Confusion Matrix of the Second Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	5	0	0	100%
MI	0	5	0	100%
AF	0	0	5	100%
Total Accuracy 100%				

4.3.3. Results of the Third Run

Table 4.10 illustrates the confusion matrix of the third run. The accuracy for the USA disease is ((4/5)*100%) which is 80%, for the MI disease it is ((4/5) * 100%) which is

80%, and for the AF disease it is ((4/5) *100%) which is 80% .This gives a total accuracy for the third run (A_{f3} = (12/15)* 100%) = 80%. A percentage of 20%, 20% and 20% from the samples of the USA, MI and AF were misclassified correspondingly.

Table 4.10: Confusion Matrix of the Third Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	4	1	0	80%
MI	1	4	0	80%
AF	1	0	4	80%
		Total Accuracy 80%		

4.3.4. Results of the Fourth Run

Table 4.11 illustrates the confusion matrix of the fourth run. The accuracy for the USA disease is ((5/5)*100%) which is 100%, for the MI disease it is ((4/5) * 100%) which is 80%, and for the AF disease it is ((5/5) *100%) which is 100% .This gives a total accuracy for the fourth run (A_{f4} = (14/15)* 100%) = 93.3%. A percentage of 20% from the samples of the MI were misclassified.

Table 4.11: Confusion Matrix of the Fourth Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	5	0	0	100%
MI	0	4	1	80%
AF	0	0	5	100%
		Total Accuracy 93.3%		

4.3.5. Results of the Fifth Run

Table 4.12 illustrates the confusion matrix of the fifth run. The accuracy for the USA disease is ((4/5)*100%) which is 80%, for the MI disease it is ((5/5) * 100%) which is 100%, and for the AF disease it is ((4/5) *100%) which is 80% .This gives a total accuracy for the fifth run (A_{f5}= (13/15)* 100%) = 86.6%. A percentage of 20% and 20% from the samples of the USA and AF were misclassified respectively.

Table 4.12: Confusion Matrix of the Fifth Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	4	1	0	80%
MI	0	5	0	100%
AF	0	1	4	80%
		Total Accuracy 86.6%		

4.3.6. Results of the Sixth Run

Table 4.13 illustrates the confusion matrix of the Sixth run. The accuracy for the USA disease is ((3/5)*100%) which is 60%, for the MI disease it is ((5/5) * 100%) which is 100%, and for the AF disease it is ((5/5) *100%) which is 100% .This gives a total accuracy for the sixth run (A_{f6}= (13/15)* 100%) = 86.6%. A percentage of 40% from the samples of the USA were misclassified.

Table 4.13: Confusion Matrix of the Sixth Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	3	2	0	60%
MI	0	5	0	100%
AF	0	0	5	100%
		Total Accuracy 86.6%		

4.3.7. Results of the Seventh Run

Table 4.14 illustrates the confusion matrix of the seventh run. The accuracy for the USA disease is ((4/5)*100%) which is 80%, for the MI disease it is ((4/5)*100%) which is 80%, and for the AF disease it is ((5/5) *100%) which is 100% .This gives a total accuracy for the seventh run (A_{f7}= (13/15)* 100%) = 86.6%. A percentage of 20% and 20% from the samples of the USA and MI respectively were misclassified.

Table 4.14: Confusion Matrix of the Seventh Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	4	1	0	80%
MI	0	4	1	80%
AF	0	0	5	100%
		Total Accuracy 86.6%		

4.3.8. Results of the Eighth Run

Table 4.15 illustrates the confusion matrix of the eighth run. The accuracy for the USA disease is ((5/5)*100%) which is 100%, for the MI disease it is ((3/5) * 100%) which

is 60%, and for the AF disease it is ((4/5) *100%) which is 80% .This gives a total accuracy for the eighth run (A_{f8}= (12/15)* 100%) = 80%. A percentage of 40% and 20% from the samples of the MI and AF respectively were misclassified.

Table 4.15: Confusion Matrix of the Eighth Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	5	0	0	100%
MI	1	3	1	60%
AF	0	1	4	80%
		Total Accuracy 80%		

4.3.9. Results of the Ninth Run

Table 4.16 illustrates the confusion matrix of the ninth run. The accuracy for the USA disease is ((5/5)*100%) which is 100%, for the MI disease it is ((4/5) * 100%) which is 80%, and for the AF disease it is ((5/5) *100%) which is 100% .This gives a total accuracy for the ninth run (A_{f9}= (14/15)* 100%) = 93.3%. A percentage of 20% samples from the MI were misclassified.

Table 4.16: Confusion Matrix of the Ninth Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	5	0	0	100%
MI	1	4	0	80%
AF	0	0	5	100%
		Total Accuracy 93.3%		

4.3.10. Results of the Tenth Run

Table 4.17 illustrates the confusion matrix of the tenth run. The accuracy for the USA disease is ((5/5)*100%) which is 100%, for the MI disease it is ((5/5) * 100%) which is 100%, and for the AF disease it is ((5/5) *100%) which is 100% .This gives a total accuracy for the tenth run (A_{10}= (15/15)* 100%) = 100%.In this run there were no misclassified samples.

Table 4.17 Confusion Matrix of the Tenth Run

Disease Type	USA	MI	AF	Disease Classification Accuracy
USA	5	0	1	100%
MI	0	5	0	100%
AF	0	0	5	100%
		Total Accuracy 100%		

4.3.11. Summary of the Results

To summarize the testing results of the ten runs, Table 4.18 illustrates the classification accuracy for all runs. As we notice the highest classification accuracy obtained in run 2 and run 10 with an accuracy of 100% and the lowest accuracy obtained was in run 3 and run 8 with accuracy of 80%. Also, the total classification accuracy was 89.3%. These results were affected by the nature of the data splits for each fold; the two folds that obtained 100% accuracy may had examples in the training sets that were very similar to the data pattern that was in the test set, so this produced high classification accuracy. But, in the two folds that obtained 80% accuracy the examples that the network

was trained with were not close to the data pattern in the test set, which produced low classification accuracy.

Table 4.18: Summary of All Classification Accuracy for All Runs

Test Sets	Classification Accuracy %
Run 1	86.6
Run 2	100
Run 3	80
Run 4	93.3
Run 5	86.6
Run 6	86.6
Run 7	86.6
Run 8	80
Run 9	93.3
Run 10	100
Average Accuracy	89.3%

Table 4.19 illustrates the experimental results of the 10 test sets that have also been presented as a confusion matrix.

Table 4.19: The confusion matrix of the diagnostic result of the heart diseases: 10- fold cross validation

Disease Type	USA	MI	AF	Row Sum
USA	44	6	0	50
MI	4	43	3	50
AF	1	2	47	50
Column Sum	49	51	50	150

Form the confusion matrix shown in Table 4.19 the accuracy for each class can be calculated as presented in Table 4.20; where the proportion of the disease type in the test dataset that have been correctly recognized by the chosen model is shown. We notice that

the MI disease obtained the lowest classification accuracy with a percentage of 86%, the highest classification accuracy was obtained by the AF with a percentage of 94%, while the USA obtained a percentage of 88% classification accuracy; giving an average accuracy of 89.3%.

Table 4.20: Proportion of correctly classified diseases

Disease Type	Accuracy %
USA	(44/50)*100% = 88%
MI	(43/50)*100% = 86%
AF	(47/50)*100% = 94%
Average Accuracy	(133/150)*100% = 89.3%

These classification accuracies for each disease can be simply explained as the following: according to the physicians, MI and USA have similar symptoms, which make the diagnosis process difficult even for an expert practitioner (Devigne-Nickens, 2007), so the obtained close accuracy results for both diseases in this thesis were realistic. Although the AF disease has some similar symptoms with the other two diseases, but it can be differentiate it from the other two diseases by some specific variables such as an irregular heart beat, raised jugular venous pressure and palpitation an AF patient may suffer from (Benjamin et al., 1998), that can be noticed in the high classification accuracy the AF obtained in this thesis in comparison with the MI and USA.

Chapter Five

Conclusion and Future Work

5. Conclusions and Future Work

Conclusions and Future Work

CVD were the first cause of death in Jordan with a percentage of 35.3% of the total deaths for the year 2007 as was shown in the mortality records for that year (Dababneh et al., 2010). Therefore, an accurate diagnosis will play a major role for the life's saving of a patient. Also, many researchers developed CDSSs to assist physicians in the diagnosis process for multiple diseases, where these systems proofed its effectiveness in obtaining high diagnosis accuracy (Yan et al., 2006) (Kayaer and Yildirim, 2003) (Kiyan and Yildirim, 2003).

Based on this, this thesis proposed a CDSS for heart disease diagnosis. The heart diseases were USA, MI and AF. The proposed system was developed using one of the most widespread machine learning techniques; a MLP feed forward neural network and trained using the Rprop training algorithm. It consisted of: an input layer with 58 neurons (representing the input variables for each patient such as, age, gender, pulse, etc.), 3 hidden layers (where the number of neurons in the first, second and third hidden layer were 29, 14 and 7 respectively) and an output layer (that produced the type of the disease the patient suffered from). Moreover, a ten-fold cross validation was used to access the generalization of the proposed system using 162 patients' medical recodes collected from the cardiology clinics at JUH. An accuracy of 86%, 88% and 94% were obtained for the MI, USA and AF respectively, giving an average of 89.3% classification accuracy.

From these results we conclude that; based on JUH clinicians' opinion the proposed system proofed its usefulness in the support of clinic diagnosis decision of heart diseases. Moreover, if more dataset was used for training the proposed system, that would give more

robust results. However, some limitations were noticed in this thesis; some medical records contained missing values because they were filled with resident doctors which made it impossible to use these records in the heart disease dataset. Also, ECG findings needed a clinician to interpret them.

In future, this work can be extended by using other types of ANNs such as, GRNN, PNN and RBF to compare the results obtained from them with the one obtained using the MLP. Also, we can apply one of the feature selection algorithms on the heart disease dataset in order to choose the medical variables that play a major role in the diagnosis process which may give more accurate results. Moreover, applying other machine learning methods such as Artificial Immune Systems, Neuro-Fuzzy and Decision Trees and comparing the results obtained with the one obtained using the ANN. Finally, since this system was built using the medical records that were collected from JUH which is a training hospital, we may apply the system on the medical records collected from other types of hospitals such as military, public and private hospitals in order to compare the results obtained with the one obtained using JUH medical records.

REFRENCES

AlTimemy.A. and Al Naima.F.(2010), Comparison of Different Neural Network Approaches for the Prediction of Kidney Dysfunction, **International Journal of Biological and Life Sciences,** 6 (2), 84-90.

Bakul, G., Tiwary, U.S. and Lahiri, T. (2009), Relative Amplitude Based Features of Characteristic ECG-Peaks for Identification of Coronary Artery Disease, **In proceeding of the First International Conference on Intelligent Human Computer Interaction**, India, 140-146

Bascil.S.,and Temurtas.F.(2009), A Study on Hepatitis Disease Diagnosis Using Multilayer Neural Network with Levenberg Marquardt Training Algorithm, **Journal of Medical Systems,** 33 (1), 1-4

Benjamin, E., Wolf, P., D'Agostino, R., Silbershatz, H., Kannel, W. and Levy, D. (1998). Impact of Atrial Fibrillation on the Risk of Death: the Framingham Heart Study. **Circulation** 98 (10), 946–52.

Brause Rüdiger W. (2001), Medical Analysis and Diagnosis by Neural Networks, **In Proceedings of the Second International Symposium on Medical Data Analysis** London, UK, 1-13.

Dababneh, f., Asad, M. and Abu Diab, A. (2010), Mortality Data in Jordan for 2007, Information and Research Newsletter (Ministry of Health, Amman, Jordan), Retrieved February, 25, 2011 from
http://www.moh.gov.jo/MOH/arabic/publications.php

Desvigne-Nickens, Patrice, (2007). Heart Disease, WomensHealth.gov, Retrievd February, 25, 2011 from
http://www.4woman.gov/faq/heartdis.htm

Duda, R., Hart, P. and Stork, D. (2001), **Pattern Classification**, (2nd ed) , NewYork: John Wiley & Sons, Inc.

Fausett, L Laurene (2001),**Fundamentals of Neural Networks**, **Architectures, Algorithms and Application**,(1st ed), New Jersey: Pearson Prentice Hall Education , Inc.

Freeman , J and Skapura , D ,(1991) , **Neural Networks Algorithms, Applications, and Programming Techniques,** (1st ed), New York: Addison-Wesley Publishing Company, Inc

Geoffrey, H. and Terrence, S. (1999) **Unsupervised Learning: Foundations of Neural Computation** (1st ed), Massachusetts: MIT Press.

Ghahramani, Saeed (2000). **Fundamentals of Probability,** (2nd ed). New Jersey: Prentice Hall

Hagan, M, Dcmuth, H, Beale, M.(1996), **Neural Network Design,** (1st ed), USA: PWS Publishing Company.

Haykin, Simon. (2001), **Neural Networks a Comprehensive Foundation,** (2nd ed), India: Pearson Prentice Hall Education , Inc.

Kayaer.K. and Yildirim.T.(2003), Medical Diagnosis on Pima Indian Diabetes Using General Regression Neural Networks**, In Proceedings of the International Conference on Artificial Neural Networks and Neural Information Processing**, Istanbul, turkey,181-184

Kiyan, T. and Yildirim, T. (2003), Breast Cancer Diagnosis Using Statistical Neural Networks, **In International XII Turkish Symposium on Artificial Intelligence and Neural Networks**, Tainn, Turkey, 754-760.

Kong, G., Xu, D. and Yang, J. (2008), Clinical Decision Support Systems: A Review on Knowledge Representation and Inference Under Uncertainties, **International Journal of Computational Intelligence Systems,** 1(2), 159-167

Krose, B and Smagt, P. (1996), **An introduction To Neural Networks**, (8th ed), Amsterdam: The University of Amsterdam.

Machado, Lucila Ohno.(1996),**Medical Application of Artificial Neural Networks: Connectionist Models of Survival**, Master's thesis, Department of Computer Science, University of Stanford, USA.

Mehrabi, S., Maghsoudloo, M. , Arabalibeik, H., Noormand, R., and Nozari, Y. (2009), Application of Multilayer Perceptron and Radial Basis Function Neural Networks in Differentiating Between Chronic Obstructive Pulmonary and Congestive Heart Failure Diseases, **Expert Systems with Applications,** 36 (3), 6956–6959.

Mehrotra, K , Chilukuri , K, Mohan and Ranka, S .(1996) ,**Elements of Artificial Neural Networks**,(1st ed), New York: Addison-Wesley Publishing Company, Inc.

Miller R. and Geissbuhler A. (2007), Diagnostic Decision Support Systems In: Berner E. (Ed), **Clinical Decision Support Systems,** (2nd ed), (pp 99-125), New-York:Springer.

Miniño, A., Xu, J.,. and Kochanek, K. (2010), Deaths: Preliminary Data for 2008 (PDF). National Vital Statistics Reports (United States: Center for Disease Control).59(2), Retrieved February, 25, 2011 from
http://www.cdc.gov/nchs/products/nvsr.html

Mitchell, Tom (1997), **MachineLearning**, (1st ed), McGraw-Hill Science.

Musen Mark A. (1997), Patient Record. In: J.van Bemmel and Mark A. Musen (Ed), **Handbook of medical informatics**, (pp 99 115), Houten: Springer.

Nakano.H., Okamoto.Y., Nakabayashi.H., Takamatsu.S., Tsujii.H., and Matsuoka.H.(1996) , Application of Neural Network to the Interpretation of Laboratory Data for the Diagnosis of Two Forms of Chronic Active Hepatitis, **International Hepatology Communications,** 5(3), (160-165).

Paulin, F. and Santhakumaran, A. (2011), Classification of Breast Cancer by Comparing Backpropagation Training Algorithms, **International Journal on Computer Science and Engineering**, 3(1), 327-332.

Ramalingam, V., Palaniappan, B., Panchanatham, N. and Palanivel, S. (2006), Measuring Advertisement Effectiveness – A Neural Network Approach, **Expert Systems with Applications,** 31(1), 159–163.

Rasheed, Zeehasham (2009), **Adaptive Fuzzy Logic Based Framework for Handling Imprecision and Uncertainty in Pattern Classification of Bioinformatics Datasets**, Master's thesis, Department of Computer Science, King Fahd University of Petroleum and Minerals, Saudi Arabia.

Riedmiller, M. and Braun ,H. (1993), A Direct Adaptive Method for Faster Backpropagation Learning: The RPROP Algorithm, **Neural Networks, IEEE International Conference, San Francisco, CA , USA** ,586-591.

Stefko, Kamil (2007), Coronary Artery Diagnosis Aided by Neural Network, **Polish Journal of Medical Physics and Engineering,** 13(3), 149-155.

Tanser, Paul (2006), Heart, Merck Manual of Medical Information, Retrieved February,25,2011 from
http://www.merckmanuals.com/home/sec03/ch020/ch020b.html

Temurtas.H., Yumusak.N., Temurtas.F.(2009),A Comparative Study on Diabetes Disease Diagnosis Using Neural Networks, **Expert Systems with Applications,** 36(4), 8610-8615.

Thammano, A., Moolwong, J.(2009),A New Computational Intelligence Technique Based on Human Group Formation, **Expert Systems with Applications**, 37(2), 1628–1634

Thornett, Andrew M. (2001), Computer Decision Support Systems in General Practice, **International Journal of Information Management,** 21(1), 39-47.

Toscano, Joseph. (2003), Angina: A Patient Guide. HeartInfo.org Retrieved February, 25, 2011 from
http://www.healthcentral.com/heart-disease/patient-guide-44616-6.html

Turkoglu, I., Arslanb,A. and Ilkayc, E. (2002), An Expert System for Diagnosis of the Heart Valve Diseases, **Expert Systems with Applications** , 23(3) , 229–236

Vahdani.P., Alavian.S., Aminzadeh.Z., Raoufy.M., Gharibzadeh.S., Vahdani.G., Fekri.S., and Eftekhari.P.(2009), Using Artificial Neural Network to Predict Cirrhosis in Patients with Chronic Hepatitis B Infection with Seven Routine Laboratory Findings, **Hepatitis Monthly ,** 9(4), 271-275

Veropoulos, Konstantinos (2001), **Machine Learning Approaches to Medical decision Making,** Doctoral thesis, Department of Engineering Mathematics, University of Bristol, United Kingdom.

Williams, M., Restieaux, N. and Low C. (1999), Myocardial Infarction in Young People with Normal Coronary Arteries, **Heart** 79 (2), 191–4.

Xu,S. and Chen, L. (2008), A Novel Approach for Determining the Optimal Number of Hidden Layer Neurons for FNN's and Its Application in Data Mining**, In the 5th International Conference on Information Technology and Applications,** Cairns, Australia, 683-686

Yan, H., Jiang, Y., Zheng, J., Peng, C. and Li, Q. (2006), A Multilayer Perceptron-Based Medical Decision Support System for Heart Disease Diagnosis, **Expert Systems with Applications**, 30(2), 272-281.

نظام لدعم القرار الطبي لتشخيص أمراض القلب

إعداد
نسرين عدنان حمد

المشرف
الدكتور محمد الشريدة

المشرف المشارك الأول
الدكتور محمود أبو عبيلة

المشرف المشارك الثاني
الدكتور معاذ الصمادي

ملخص

تعتبر أمراض القلب والأوعية الدموية المسبب الأول للوفيات بالعالم ولا سيما في الأردن، حيث أن سجلات الوفيات للعام 2007 أظهرت أن المسبب الرئيسي للوفيات بالأردن هي أمراض القلب والأوعية الدموية بنسبة 35.3%. لذلك يعتبر التشخيص الدقيق لأمراض القلب عاملا حاسما في إنقاذ حياة المرضى.

لقد اتجه الكثير من الباحثين في العقود الأخيرة إلى تطوير أنظمة لدعم القرارات الطبية، حيث أن الهدف من هذه الأنظمة مساعدة الطبيب في عملية تشخيص الأمراض وبالتالي التقليل من نسبة الخطأ الطبي الممكن حدوثه في عملية التشخيص. بناءا على ذلك تم اعتماد الكثير من المنهجيات والوسائل لتحقيق هذا الهدف. أحد أشهر هذه الوسائل والتي تنتمي لعائلة الحسابات الذكية هي: الشبكات العصبية الاصطناعية المستوحاة فكرتها من الأنظمة العصبية البيولوجية.

في هذه الرسالة، تم تصميم نظام لدعم القرار الطبي لتشخيص ثلاث أمراض للقلب بواسطة شبكة عصبية اصطناعية متعددة الطبقات ذات تغذية أمامية. تستقبل طبقة الإدخال 58 متغير مدرجين ضمن ست مجموعات وتم ترميزهم بالطريقة المقترحة بهذه الرسالة. في حين أن طبقة الإخراج ستنتج نوع المرض الذي يعاني منه المريض.لقد تم استعمال طريقة تكرارية لتحديد عدد الطبقات المخفية و عدد الخلايا العصبية في هذه الطبقات. لتدريب الشبكة تم استعمال خوارزمية التدريب (Rprop). بالإضافة إلى ذلك تم تطبيق طريقة المجموعات العشر للتحقق من مقدرة الشبكة العصبية على التعامل مع بيانات جديدة لم تتعامل معها في مرحلة التدريب. لقد تم تدريب وإجراء الاختبارات على النظام بواسطة 162 سجل طبي تم الحصول عليهم من عيادات القلب في مستشفى الجامعة الأردنية. حيث أظهرت التجارب بأن النظام المقترح في هذه الرسالة استطاع تشخيص الثلاث أمراض للقلب بدقة 89.3%.